User-Perceived Quality Assessment for VoIP Applications

Technical Report

Razvan Beuran **Mihai Ivanovici**

January 2004

The authors are currently Project Associates with
CERN, 1211 Genève 23, Switzerland.

They can be reached using their corresponding e-mail addresses:
{Razvan.Beuran, Mihail.Ivanovici}@cern.ch.

Table of Contents

Abstract

We designed and implemented a system that permits the measurement of network Quality of Service (QoS) parameters. This system allows us to objectively evaluate the requirements of network applications for delivering user-acceptable quality. To do this we compute accurately the network QoS parameters: one-way delay, jitter, packet loss and throughput. The measurement system makes use of a global clock to synchronise the time measurements in different points of the network. To study the behaviour of real network applications specific metrics must be defined in order to assess the user-perceived quality (UPQ) for each application.

Since we measure simultaneously network QoS and application UPQ, we are able to correlate them. Determining application requirements has two main uses: (i) to predict the expected UPQ for an application running over a given network (based on the corresponding measured QoS parameters) and understand the causes of application failure; (ii) to design/configure networks that provide the necessary conditions so that an application will run with a desired UPQ level.

This document presents the work we carried out in order to determine the relationship between network QoS and UPQ for voice over IP (VoIP) applications. For VoIP we measured the UPQ using the PESQ score (Perceptual Evaluation of Speech Quality). We performed uni-directional tests, which focus on the perceived quality of the speech itself depending on packet loss and jitter. Voice data is transmitted encoded; the codecs we tested are: G.711, G.726, GSM and G.729.

We concluded that G.729 is to be the most robust codec in the range of network conditions under study. It provides almost the same acceptable perceived quality throughout the studied loss-jitter space. The codec G.711 performs better than the other codecs as long as network conditions are good (loss rate smaller than 3% and jitter below 20 ms). G.726 gives results that are better at most 7% than those for GSM; however this is achieved at the expense of a transmission rate which is 2.5 times larger.

1 Introduction

Quality of Service (QoS) has become more widely recognised as an important issue since network applications with real-time requirements have started to spread on a larger scale. At the moment there are not many systems able to correlate the QoS provisioned by networks with the user-perceived quality (UPQ) for specific applications, such as voice over IP (VoIP) or file transfer. Knowing the requirements of such applications allows predicting whether a certain connection is valid for a certain application and what is the expected perceived quality for that application.

The system we designed and implemented permits the measurement of network Quality of Service (QoS) parameters. This system allows us to objectively evaluate the requirements of network applications for delivering user-acceptable quality. We make use of FastEthernet taps to monitor full-duplex traffic and programmable network interface cards to extract the information needed to compute the network QoS parameters: one-way delay, jitter, packet loss and throughput. The measurement system makes use of a global clock to synchronise the time measurements in different points of the network.

This document presents the work we carried out in order to determine the relationship between network QoS and UPQ for VoIP applications. We focused on VoIP because it is currently one of the most used network applications in which user perception plays a fundamental role. VoIP applications don't stress networks from the point of view of bandwidth, because the requirements of speech transmission are low (e.g. 64 kbps of voice data or even less). However delay (including delay variation) and packet loss are very important because they affect directly the perceived quality and hence user satisfaction. This means that VoIP is one of the applications that could benefit greatly from using QoS control techniques and service differentiation. One of the goals of our research is to provide the information required so that one can tune networks in order to offer the desired application-level quality to users.

We quantify the relationship between network QoS and UPQ for VoIP applications by simultaneously measuring the QoS in the network and assess the UPQ for VoIP communication using the Perceptual Evaluation of Speech Quality (PESQ) score [P.862]. Thus we are able to correlate network conditions with the UPQ for this application, which allows the performance of two main tasks:

- predicting the expected UPQ for a VoIP application running over a given network taking into account the corresponding measured QoS parameters; understanding the causes of application failure by defining minimum requirements that must be met by the network;
- designing/configuring a network to provide the necessary conditions for a VoIP application to run at a desired UPQ level.

Most network performance evaluation is currently done either through experimental work in real networks, simulation or an analytical approach. Emulation is a hybrid performance evaluation methodology enabling controlled experimentation with real applications. This is an integral component to our approach of studying QoS.

1.1 State of the art

There are various projects involved in the field of QoS, especially related to service differentiation. For example TEQUILA's objective is to study, implement and validate network service definition and traffic engineering tools built upon DiffServ in order to obtain quantitative end-to-end QoS guarantees [God-00], [Gri-00], [Man-01]. QBone specifies and deploys the QBone Premium Service, a virtual leased-line IP service built also on DiffServ forwarding primitives [Tei-99]. Internet2 has also started the End-to-End Performance Initiative [Int2], whose objective is to create a predictable and well-supported network environment. All these projects focus on network QoS mechanisms, whereas we undertook first establishing application requirements such that user expectations are fulfilled. Only subsequently can QoS mechanisms be deployed to meet these requirements.

The relationship between network conditions and application performance has been studied by a number of projects. Internet2 QoS Working Group has published a survey on QoS application needs [Mir-02]. TF-STREAM reported on best-practice guidelines for deploying real-time multimedia applications [Cav-02]. HEAnet reviewed several aspects of perceived quantitative quality of applications [Rei-**]. Generally, these approaches are qualitative – we aim to create a quantitative representation of UPQ that can be related to QoS parameters.

There is a number of studies on computer networks from a VoIP performance perspective. In [Jia-03] the authors evaluate the availability of VoIP services typically achieved in the current Internet; the quality of the communication itself is however not taken into account. Various traffic measurements are used in [Bou-02] to quantify the impact of link failures on VoIP performance. This is done through the use of the E-model [G.107], for whose R-factor no complete standard analytical expression exists. The authors of [Mar-02] make also use of the E-model (in a more fine-tuned version) to assess VoIP quality over Internet backbones. Still measurements are performed over real networks, hence do not provide a complete view on the behaviour of VoIP in the full range of possible network conditions. [Con-02] uses simulation to get closer to this goal, but this leads to the loss of finer details which only show up when real implementations are used to send and receive packets. To assess perceived VoIP quality the authors use a variant of PSQM.

The novelty of our approach consists in three main items:

- the use of a custom-built system to perform accurate measurements of network QoS parameters;
- the employment of a network emulator and a real VoIP application in order to characterise a full range of network conditions;
- the utilisation of the most recent ITU-T metric for VoIP UPQ assessment, the PESQ score [P.862].

1.2 Report structure

The document is structured as follows. Chapter 2 introduces several fundamental concepts related to VoIP applications and several methods of assessing user-perceived quality for VoIP. We also discuss some issues related to the network QoS parameters varied in our tests, one-way delay variation and packet loss.

Chapter 3 presents details related to our test setup and the assessing of the software tools we used to perform VoIP-specific experiments. We also describe the preliminary tests we ran as well as all related practical details about our experiments. The chapter ends with issues related to the representation of experimental data.

Experimental results for the four studied voice codecs are presented in Chapter 4. This is followed by a thorough comparison of the codecs and a classification of them based on a criterion defined by ourselves.

The main conclusions of our study are summarised in Chapter 5.

Our report ends with acknowledgements, a glossary and a list of references.

2 Principles of VoIP

This chapter starts with several basic principles related to VoIP and its utilisation. Then we present briefly the metrics that can be used to calculate the predicted UPQ for VoIP applications.

In addition we analyse several issues related to the two parameters we varied in our tests (one-way delay variation and packet loss). When setting out to assess VoIP applications we realised there are several issues that are not fully documented in the literature. To give only a few examples: everybody measures jitter, but there is a number of possible definitions of it and they have to be specified when using the results. Dejittering is also a potentially complex task, but there are so many algorithms to do it that selecting a basic one is not easy. Moreover dealing with packets that arrive out-of-order is a problem that can be tackled in different ways.

2.1 Basic facts

Communicating via speech over the Internet (i.e. Internet telephony) is currently one of the most widely used network applications that involves the human perception in a direct manner. Voice over IP or VoIP are the terms used in general for referring to this service. Although the bandwidth requirements of speech transmission are relatively low, usually below 64 kbps of voice data, its interactivity implies high sensitivity to delay and jitter. Packet loss has also a significant influence on VoIP performance.

The reason for VoIP being an interesting application from our point of view is that with a minimum effort/expenses one can use the existing network infrastructure to make essentially free phone calls. However, in order to have users widely accept this alternative to the traditional switched telephone networks, it is mandatory that the quality of the services offered be equivalent.

One of the main differences between computer networks and telephone networks is that the first ones are packet-switched while the others are circuit-switched. In computer networks the applications running at the end nodes generate packets that are sent over the network. Although conceptually these packets are part of the corresponding logical streams of each application once they reach the switching fabric they are treated independently. For each packet routing decisions are taken and the packet is forwarded to its destination. Therefore voice packets will share the network (and the associated resource) with other application packets and each of them will have to be independently directed to its destination.

On the other hand, since telephone networks are circuit-switched, once the connection is established the circuit is reserved for that communication. This means that a certain amount of resources is allocated for the corresponding circuit, so there is no interference from the other communications. Moreover once the circuit is allocated it is certain that there will be no interruption and no significant variation in quality for the entire duration of the communication.

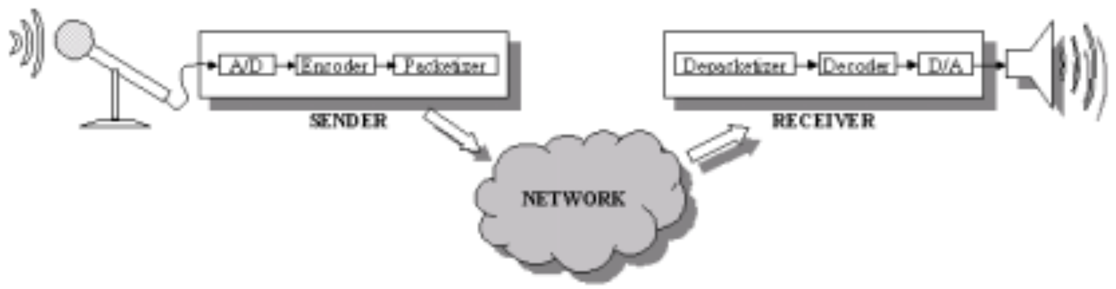

Figure 1: End-to-end data path for VoIP communication.

In Figure 1 is presented a description of an end-to-end path as needed for VoIP communication. An audio input device, such as a microphone, is required at the sending end. The audio signal is transformed to digital form by an analog-to-digital converter. Due to the packet-switched nature of computer networks, voice data has to be packetised and encoded prior to being transmitted. Encoding (as well as decoding) is done by codecs that transform sampled voice data into a specific network-level representation. Most of the codecs are defined by standards of the International Telecommunication Union, the Telecommunication division (ITU-T). Each of them has different properties regarding the amount of bandwidth it requires but also the perceived quality of the encoded speech signal. The codecs we tested are G.711, G.726, GSM and G.729.

After binary information is encoded and packetised at sender level, packets encapsulating voice data can be sent over the network. As mentioned previously, voice packets interact in the network with other application packets and are routed through shared connections to their destination.

At the receiver end packets are decapsulated and decoded. Decoding may include other steps as well, the most typical being dejittering. Other examples are error correction and packet loss concealment. The flow of digital data is then converted to analogue form again and played at an output device, usually a speaker.

Please note that for a bidirectional VoIP communication the same path exists in the opposite direction. The only interaction between these paths takes place if echo cancellation is employed.

There exist at the moment IP phones which are similar in shape with the regular telephones but instead of being connected to a phone socket they are plugged into a network connection. Therefore the act of making the phone call using VoIP can be identical to that of using regular phones*. The quality of the communication itself can be different however and it is the most important aspect of the transition from standard telephone networks to Internet telephony.

One reason for such a transition is that VoIP communication is more flexible than standard telephony. By making the appropriate choice for the codec one can control the amount of bandwidth required and one determines the intrinsic associated quality. Moreover, by managing properly the network one can use the codec which provides the desired quality level, since resources in computer networks can be allocated in different manners for various uses.

However, since the communication channel is not reserved but shared with other

* Note however that usually people sit in front of their computers and use headsets when making VoIP calls.

applications, voice packets can arrive at the receiver with a different inter-packet gap than they had at the sender, out of order, and some of them can even be lost. Assessing the relationship between precisely these factors, as quantified by means of network QoS parameters, and the UPQ of VoIP communication constitutes the scope of our research.

2.2 UPQ assessment for VoIP

Modern telecommunication networks provide a large set of voice services using many transmission systems. The rapid deployment of digital technologies in particular has lead to an increased need for evaluation of the transmission characteristics of new communication equipment.

ITU-T has defined several standards that allow an evaluation of the quality of voice communication which shall be described briefly, in chronological order. The first of them was a subjective metric, but successive attempts have been made to define objective metrics as well.

2.2.1 Mean Opinion Score (MOS)

In 1996 ITU-T has defined the methodology of determining how satisfactorily given telephone connections may be expected to perform [P.800]. The methods are intended to be generally applicable for any possible form of degradation: loss, circuit noise, transmission errors, environmental noise, talker echo, distortion due to encoding etc.

The evaluation procedure is based on subjective tests in which quality is graded by human experimenters. The following values are assigned depending on the quality of the connection:

$$Excellent = 5\,;Good = 4\,;Fair = 3\,;Poor = 2\,;Bad = 1. \qquad (1)$$

The distinction between mean conversation-opinion score (MOS_C) and mean listening-opinion score (MOS_L) is made. In the second case only the intrinsic quality is taken into account, whereas the first case includes the experimenter's opinion about the level of interactivity.

2.2.2 Perceptual Speech Quality Measure (PSQM)

As a next step, ITU-T standardised in 1998 an objective method for quality measurement of telephone band speech codecs called Perceptual Speech Quality Measure (PSQM) [P.861]. This method was initially developed by KPN, Netherlands. After comparing it to several others metrics, ITU-T concluded that the output of PSQM is best correlated with the subjective quality of coded speech.

PSQM is intended to be used for listening-only tests. It can be used for codecs with rates higher than 4 kbps, but there is insufficient information regarding its performance with respect to several factors, such as transmission channel errors, and is not intended to be used in conjunction with delay, for example.

PSQM uses psychophysical[*] representations that match the internal human representation of speech signals as closely as possible. A zero value PSQM means no impairment was present, while PSQM equal to 6.5 means a totally unusable channel.

* Psychophysical = of or relating to psychophysics, a branch of psychology concerned with the effect of physical processes (as intensity of stimulation) on the mental processes of an organism.

PSQM values can be mapped onto a MOS-like scale in order to obtain an estimate of the subjective quality.

2.2.3 The E-model

The E-model appeared in 2000 and is a computational model for use in transmission planning [G.107]. This is a transmission rating model that can be used to help ensure that users will be satisfied with end-to-end transmission performance. The model integrates the impairment factors that affect communication equipment, including delay and low bit-rate codecs. These impairments are computed based on a series of input parameters for which default values and permitted ranges are specified. These should be used if the corresponding impairment situation occurs.

The MOS score (equivalent to the mean conversation-opinion score MOS_C from [P.800]) in the scale 1 to 4.5 can be obtained from the R-factor using the following formulae:

$$\text{For } R < 0: \qquad MOS = 1 \tag{2}$$

$$\text{For } 0 \leqslant R \leqslant 100: \qquad MOS = 1 + 0.035R + R(R-60)(100-R)7 \cdot 10^{-6} \tag{3}$$

$$\text{For } R > 100: \qquad MOS = 4.5 \tag{4}$$

The graph of the dependency of MOS on R is shown below. Note that the maximum obtainable MOS is 4.5, the average score that usually results from subjective tests for excellent quality, since experimenters' grades are known to vary between 4 and 5 in such conditions.

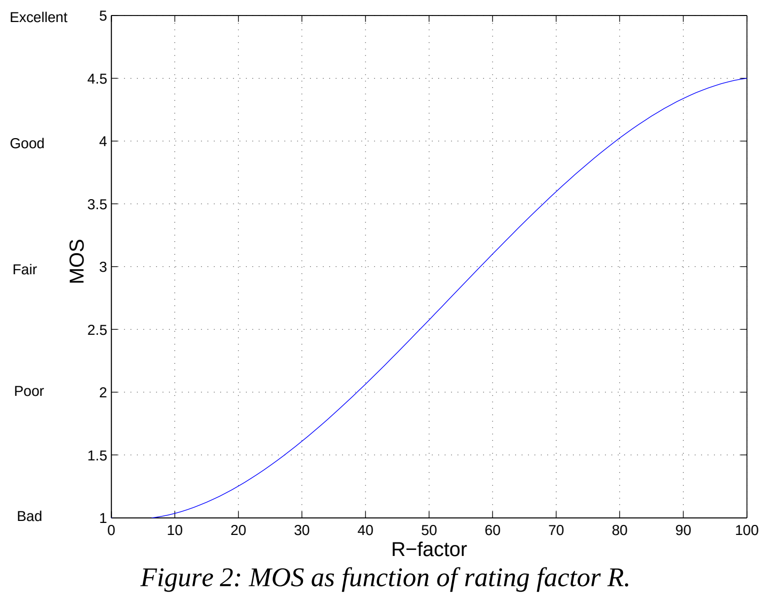

Figure 2: MOS as function of rating factor R.

A guide of the relationship between user satisfaction and R-factor is given in the table below:

R-factor (lower value)	***MOS (lower value)***	***User satisfaction***
90	4.34	Very satisfied
80	4.03	Satisfied
70	3.60	Some users dissatisfied
60	3.10	Many users dissatisfied
50	2.58	Nearly all users dissatisfied

Table 1: A guide for the relation between R-factor, MOS value and user satisfaction.

2.2.4 Perceptual Analysis/Measurement System (PAMS)

PAMS is a method developed by British Telecom (and hence subject to copyright) of determining the expected speech quality of a transmission system across any network, including those subject to packet loss or variations in delay [PAMS].

The PAMS computation process uses a model that combines a mathematical description of the psychophysical properties of human hearing with an analysis technique that takes into account the error subjectivity in the received signal from a human perception point of view. The PAMS process compares the original and degraded signal and determines MOS predictions, on a scale from 1 to 5, for the listening quality and listening effort. The score calculated by PAMS is said to be within one half of typical MOS determined by a controlled subjective test in a laboratory.

Extensive subjective tests employing human subjects across a variety of speech carrier technologies were performed to validate the PAMS model. This algorithm has been verified against a wide range of conditions including low bit-rate codecs.

2.2.5 Perceptual Evaluation of Speech Quality (PESQ)

In February 2001 ITU-T has defined the PESQ score [P.862], which is an objective method for predicting the subjective quality of narrow-band telephony and speech codecs. PESQ combines the best of PSQM and PAMS (as a result of being produced jointly by their respective developers KPN and British Telecom). In addition to PSQM, PESQ takes into account filtering, variable delay, coding distortions and channel errors.

PESQ has been thoroughly tested and has demonstrated acceptable accuracy for the following applications: codec evaluation and selection, live network testing using digital or analogue connection to the network, testing of emulated and prototype networks.

The key process in PESQ is the transformation of both the original and degraded signal into representations analogous to the psychophysical representation of audio signals in the human auditory system.

The PESQ score is mapped to a MOS-like scale, a number in the range of -0.5 to 4.5, although for most cases the output range will be between 1.0 and 4.5, the normal range of MOS values found in subjective listening quality experiments.

According to [Ser-01] the relationship between PESQ scores and audio quality is the following :

- PESQ scores between 3 and 4.5 mean acceptable perceived quality, with 3.8 being the PSTN[*] threshold – this will be termed as *good quality*;
- Values between 2 and 3 indicate that effort is required for understanding the meaning of the voice signal – this will be named *low quality*;
- Scores less than 2 signify that the degradation rendered the communication impossible, therefore the quality is *unacceptable*.

Being the most recently developed metric, PESQ outperforms the previous ones, such as PSQM or PAMS. Unlike the E-model, it doesn't require any knowledge regarding the network and uses only the original and degraded signal to compute the PESQ score. Therefore this is the best choice for our experiments, that make use of a network emulator. To compute the PESQ score we used in our research an implementation supplied by Malden Electronics Ltd. [Malden].

2.3 Delay issues

Mouth-to-ear (end-to-end) delay is an important parameter for VoIP communication from the point of view of human perception, but it doesn't affect by itself the quality of the voice signal. Hence we did not include the precise study of the influence one-way delay has on perceived quality within our research.

Note however that there are several results and ITU-T recommendations regarding influence of the end-to-end delay on UPQ [Rei-**][Y.1541]. Thus, a delay between 0 and 100-150 ms ensures high interactivity, while a delay between 100-150 and 400 ms provides an acceptable level of interactivity. Delays exceeding 400 ms are not acceptable for VoIP communication. The ITU-T metric that takes delay into account when measuring the quality of VoIP communication is the R-factor, but most of them – including the one we used – only evaluate listening quality (see Section 2.2 for details about VoIP UPQ assessment). Hence the aforementioned recommendations should be used to estimate the degree of interactivity for all metrics except the R-factor.

On the other hand, jitter (i.e. delay variation) is an important factor that has a direct influence on VoIP quality. Jitter affects packets by changing the distribution of the moments packets arrive at the receiver with respect to the distribution they had at the sender (for VoIP this distribution is deterministic: packets leave with a constant inter-packet gap). In order to counter the effects of jitter, all VoIP applications use a dejittering buffer to try to restore the initial distribution at the expense of adding a supplementary playback delay.

We must state first what is the definition of jitter used in our computations. Although ITU-T has given a definition of jitter in [I.380], we prefer the definitions from IETF in the context of IP Performance Metrics [Dem-02]. The reason is that they agree better with our view on the influence of jitter on applications. The corresponding formula (5) measures the variation of the one-way delays of consecutive packets. Given that packets leave at a constant rate, formula (5) also measures the variation of

* PSTN stands for “Public Switched Telephone Network”.

the time at which packets arrive at the receiver (since packets are equally spaced at the sender, the variation of the inter-packet gaps at the receiver is equal to the variation of the corresponding one-way delays). We claim that this is what an application "perceives" as jitter, since it has no direct knowledge about the variation of one-way delays, but it is affected by the variation of the inter-packet gaps. The formula we used for the computation of the average jitter *J* is:

$$J=\frac{1}{N-1}\sum_{i=2}^{N}|D_i-D_{i-1}|, \tag{6}$$

where *N* is the total number of VoIP packets and D_i is the one-way delay for the packet with index *i*, with $i=\overline{1,N}$.

2.3.1 Special jitter issues

As mentioned before, to limit the effects jitter has on quality, a dejittering buffer is used by all VoIP applications. In our experiments we have also tested in 0 ms buffer conditions, to illustrate how jitter affects quality when no dejittering takes place at the receiving end.

In our tests the dejittering buffer is integrated in the post-processing program "process_voice_data", so we can use exactly the same network data and change dejittering buffer parameters to examine their effects on UPQ.

For VoIP traffic the stream of data is of Constant Bit Rate (CBR) type, i.e. packets leave from the sender at a fixed rate and with a constant inter-packet gap. Applying jitter to such a stream means varying the latency of the packets so that they arrive at the receiver with a variable inter-packet gap. In our case this is done by means of the NIST Net network emulator [NIST].

The behaviour encountered most often in practice is that latency both increases and decreases during the VoIP communication, due to various changes in the network (e.g. network queue occupancy varies). This means that the differences between the values of the latency for successive packets are both positive and negative. In some cases, when the system has not reached yet steady state, it is possible to observe values of the latency that keep increasing/decreasing during communication (i.e. differences of latency values are always positive/negative). We claim that this is highly unlikely to occur, therefore we take into account the first scenario.

According to Appendix 1 this means that there exists the possibility of having out-of-order packets if average jitter exceeds a certain threshold (which is related to the inter-packet gap). Reordering is thus mandatory, since the inter-packet gap in our case was either 40 or 80 ms and the average jitter went up to 75 ms. Packet reordering also occurs in reality due to various reasons, such as alternative routes used for load balancing or routing equipment with multiple paths from input to output (e.g. Juniper routers have 4 internal paths from one input port to the same output port).

To deal with reordering there are two alternatives:

1. Deal with out-of-order packets before dejittering. This implies forcing the data to "arrive" in-order at the dejittering module by swapping packet contents if necessary before dejittering emulation and keeping the arrival times. However this only works for low jitter, otherwise the statement proven in Appendix 1 holds and the average jitter is limited by the inter-packet gap;

2. Deal with out-of-order packets while dejittering. In this case reordering is done “on the fly” when emulating dejittering (by using the included timestamps when selecting packets to be played). This method can be used for any values of the jitter.

Due to the limitations of solution 1 reordering is done in our experiments according to solution 2 (see next section for details).

2.3.2 Dejittering buffer

The simplest approach when receiving VoIP packets is to play the available packets at their corresponding playback time. The perceived quality of such a rendering is however very low, since in many cases packets arrive too early or too late with respect to the time at which they should be played (see for example Figure 12 on page 33 for a quantification of this effect; note that no dejittering is equivalent to a zero playback delay). In order to counter the negative effect of jitter one has to perform dejittering, i.e. delay playback so that packets can be played with the fixed rate they were sent at.

There are two main classes of dejittering algorithms: static and dynamic. Static dejittering implies that there is a fixed playback delay, which may be configured in the beginning of the communication, but remains fixed throughout. Some consider that performance can be improved by varying the playback delay. This makes the assumption that past behaviour is an indication of future behaviour, which is not always true.

Dynamic dejittering uses variable playback delays for each packet with the goal of optimising quality by adaptation to network conditions [Moo-98]. A variety of algorithms exists for dynamic dejittering. Playback delay is adaptively adjusted to be the smallest delay so that playback quality is maximised (which is generally equivalent to avoiding excessive packet loss due to the arrival of packets at the receiver after their playback time). These algorithms track the arrival of recently received packets and use this information to predict the arrival of the next packets using various estimation techniques.

For the work we carried out we considered that first of all a baseline is needed, to indicate what a worst-case behaviour of the system is. Then more complex situations can be taken into account and potential improvements evaluated.

Therefore the dejittering buffer strategy we use is the simplest possible. The playback time of each packet that arrives is delayed by a constant amount of time, the *PLAYBACK_DELAY*. The next packet to be played is selected from the packets in the buffer according to timestamps. If no packet with the corresponding playback time exists in the buffer, then a period of silence with the duration of a packet is created. Potential reordering in the network is accounted for by this method due to the use of timestamps.

Note that all incoming packets are placed in the dejittering buffer regardless of their arrival and playback times, but packets whose playback time has expired are discarded. This is called “dejittering loss” and is the way in which jitter affects voice quality. The algorithm for playback is described below in pseudo-code:

```
1. Initialise the first playback time
2. Repeat until there are no more packets
   (a) Select that packet in the dejittering buffer whose timestamp equals the current
       playback time
       i.  If this packet exists, play the packet
       ii. If no such packet is available, insert an equivalent silence gap
   (b) Compute the playback time of the next packet
```

The main input parameter of the dejittering algorithm is called *PLAYBACK_DELAY*, which is the amount of time playback is delayed by in order to ensure smooth playback. Since we use static dejittering, *PLAYBACK_DELAY* remains constant for the entire duration of each experiment. We decided upon its value following some preliminary tests.

There are various ways to change this simple algorithm. For example there is an alternative solution to dropping packets that have arrived too late. This solution implies accelerating playback such that the delayed audio data lasts less time and therefore there is no time shift of the entire voice sequence. There is also the possibility to play the voice signal at a lower rate, thus minimising the intervals of silence. Evaluating the potential improvement due to these changes can be done using PESQ scores, but is not within the scope of our research. Our tests provide a baseline for VoIP performance and we view such improvements only as “add-ons”. Since what we used is the simplest possible algorithm, the results obtained represent worst-case behaviour.

2.4 Packet loss issues

There are several issues related to packet loss. One of them is the way in which packets are lost in NIST Net. At the moment loss events are not correlated, but the influence of a possible correlation has not been studied. This is potentially interesting because in practice packet loss is mainly caused by congestion and many times a sequence of packets is lost. Evaluating these situations is however difficult and was not part of our research.

Another issue is Packet Loss Concealment (PLC). This refers to the actions taken when packets are lost. The straight forward method is to simply insert a silence gap instead of the lost packet, with an equivalent duration. A better solution is to artificially create voice samples based on previous and possibly following samples that are already present in the dejittering buffer.

Clearly using something closer to the signal that was lost instead of silence improves quality. There is a plethora of PLC algorithms, most of them use linear prediction techniques such as [Güz-01]. A survey of various schemes can be found in [Wah-00]. However analysing this solution requires implementing a packet loss concealment algorithm, which is out of the scope of our current research. According to [Güz-01] we can say that improvement is between 0.5 and 1.5 on the MOS-PESQ scale, depending on the loss rate and packet size (obviously losing small packets is easier and better masked, since they contain less audio data). For 40 ms packets the reported improvement doesn't exceed 0.6 on the MOS-PESQ scale.

3 Experimental Apparatus

In this chapter we present the setup used in our experiments as well as the two main software tools that were used in our tests: the NIST Net network emulator and the Speak Freely VoIP application. For NIST Net we had to evaluate its capabilities and see if our expectations regarding the types of degradation and the corresponding precision are met. The same holds for Speak Freely, which had to be modified in order to provide all required functionality (e.g. saving the received voice signal to a file).

We also present a series of preliminary tests performed in order to calibrate the system, the details of the experiments and a couple of issues related to result representation.

3.1 Test setup

The setup we used in our experiments is shown in Figure 3:

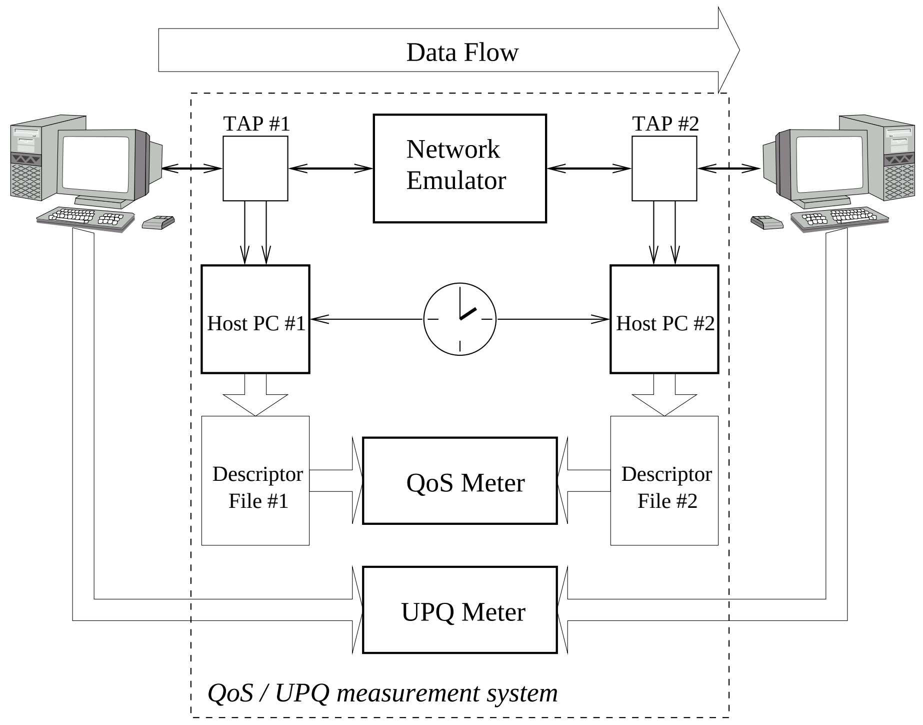

Figure 3: VoIP test setup.

We remind here the basics of the setup, but a more detailed description of the entire system and its components can be found in [Beu-03].

When testing VoIP communication, a VoIP application (SpeakFreely in our case) runs on the end PCs generating a VoIP data flow based on an input voice recording file. This data is sent through the NIST Net network emulator that artificially introduces packet loss and delay allowing us to study application behaviour under various network conditions.

The QoS/UPQ measurement system includes FastEthernet network taps that monitor network traffic in both directions (however our tests only use uni-directional VoIP data flows). Programmable NICs hosted by two PCs are utilised to extract from the mirrored traffic the essential information required to compute the QoS parameters. This information is stored as descriptor files. The descriptors files are used by the QoS Meter software to compute off-line all the studied QoS metrics. Custom-built PCI clock cards are employed to synchronise the time measurements at the two test points.

Our VoIP application produces an output file that allows the reconstruction of the received voice signal. Based on this file and the initial file sent by the VoIP application, the UPQ Meter software estimates the perceived quality for the communication (using the PESQ score).

3.2 NIST Net network emulator

There are three main ways to take when performing network related experiments. The first one is simulation, where a representation of the real network is created and then the simulation engine uses the models associated to the simulated network elements in order to compute the estimated results for the studied links. The second approach is to use real network applications running over real networks. In this way accurate measurements of what happens in a real scenario can be performed. The problem in this case is the fact that a real network cannot be thoroughly and precisely controlled, therefore the range of scenarios that can be tested is limited.

There exists also a hybrid approach that combines the advantages of the previous methods. This approach implies the use of a network emulator, a tool that reproduces the effects real networks have on traffic by introducing artificial degradation of the QoS parameters. The traffic is still real traffic, so that one can still analyse the effects network conditions have on real applications. However, since the degradation introduced by the emulator is controllable, a wide range of network conditions can be studied.

3.2.1 Features

The NIST Net network emulator [NIST] is a freeware tool from the National Institute of Standards and Technology. Using it we are able to artificially introduce packet loss and delay at rates up to 100 Mbps with minimum-size packets.

Intended packet loss is given to NIST Net as a percentage of lost packets. The possibility of using a correlation between successive loss events exists, which means that there is a non-negligible probability to lose more packets in a row. We decided not to enable this correlation since our purpose is to have a baseline, i.e. study the most basic scenario. Hence in all our tests the corresponding parameter was set to zero, i.e. no correlation was present.

For delay one supplies the intended average delay and the standard deviation (which is a measure of the jitter). The possibility of using also a correlation between successive delay values exists, that is successive delay values will not differ significantly. Our basic scenario assumes delay values are not correlated, therefore for all our tests this parameter was set to zero.

3.2.2 Evaluation

Using the QoS/UPQ measurement system described in [Beu-03] we measured the actual values of the QoS parameters as degraded by NIST Net. This is done with an accuracy of 1 μs for the one-way delay. All our results (including the following graphs) are based on the actual values of the QoS parameters, and not the intended ones as supplied to NIST Net; therefore we have a very accurate description of the traffic flows.

We performed some tests to see what is the delay distribution. According to the documentation of NIST Net and the source code, the emulator uses as the default an experimentally observed distribution*. This is shown in Figure 4 for an average delay of 300 ms and a standard deviation (average jitter) of 20 ms. Note the long-tail shape of the distribution and the fact that the delays are limited to 380 ms, hence the small peak at the end. This range of latency values, given by the formula below, comes from the design of the NIST Net artificial delay algorithm:

$$Delay \in [AvgDelay - 4 \cdot StdDev , AvgDelay + 4 \cdot StdDev]. \qquad (5)$$

Figure 5 shows the same data in a different representation, using the computed Cumulative Distribution Function (CDF) of the delay, as 1 – CDF(one-way delay) on a logarithmic scale**. This allows making a determination of the number of packets that have a latency higher than a certain limit (e.g. in our case 40% of the packets have latencies exceeding 300 ms).

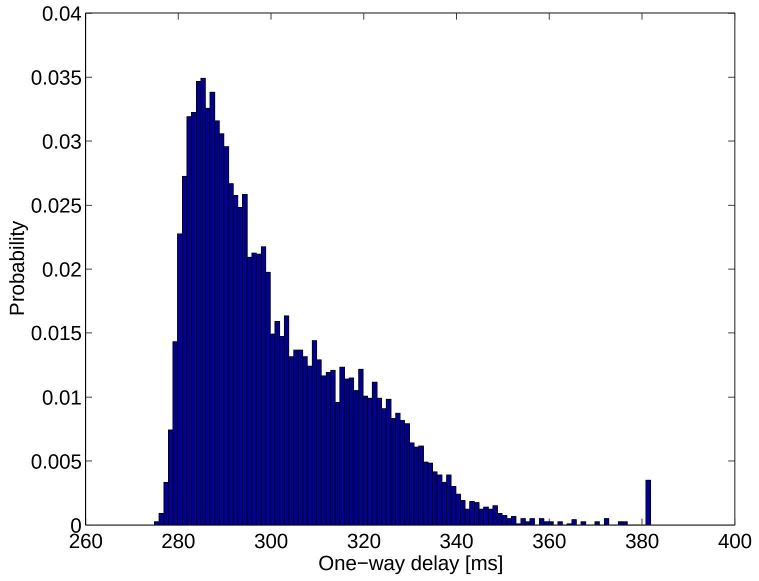

Figure 4: One-way delay histogram (average one-way delay = 300 ms, average jitter = 20 ms).

* The distribution is based on 25,000 `ping` roundtrip times to a “distant” point (www.ntt.co.jp).

** This means that the values on the *y* axis (1, 0.9, 0.8 etc.) are not represented on a linear scale, but according to their logarithm.

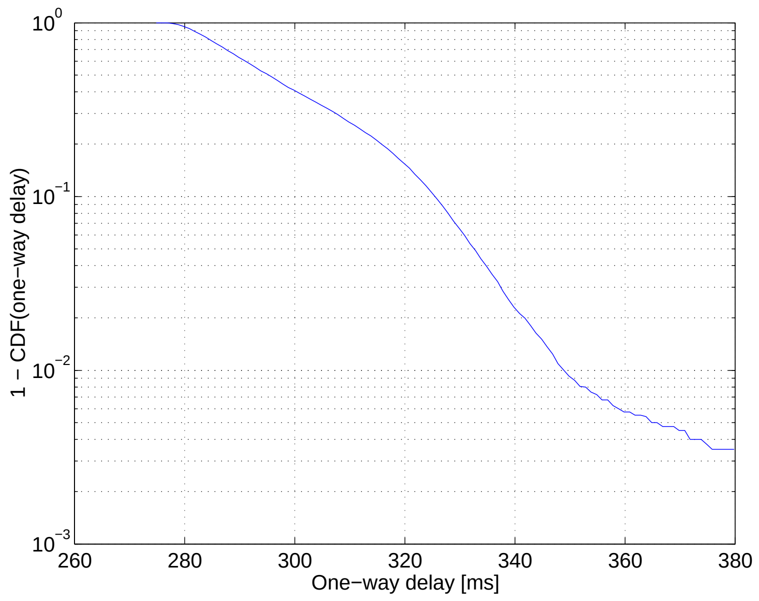

Figure 5: 1 – CDF (one-way delay) (average one-way delay = 300 ms, average jitter = 20 ms).

We computed jitter using the definition in Section 2.3 and its histogram is represented in Figure 6. In order to better emphasise the properties of the introduced jitter we computed its CDF and plotted 1 – CDF(jitter) on a logarithmic scale (see Figure 7). From this plot one can determine, for example, that about 10% of the instantaneous jitter values are above 40 ms and 0.6% above 80 ms. This means that by compensating a jitter of 80 ms (using a dejittering buffer), 99.4% of the packets will be played normally, i.e. in order and without gaps, while the rest will cause silence gaps to appear because of non availability when needed for playback.

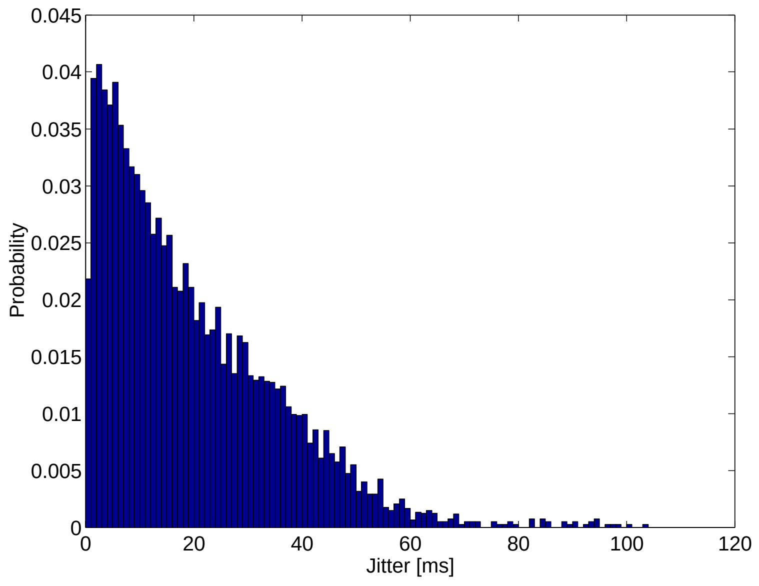

Figure 6: Instantaneous jitter histogram (average one-way delay = 300 ms, average jitter = 20 ms).

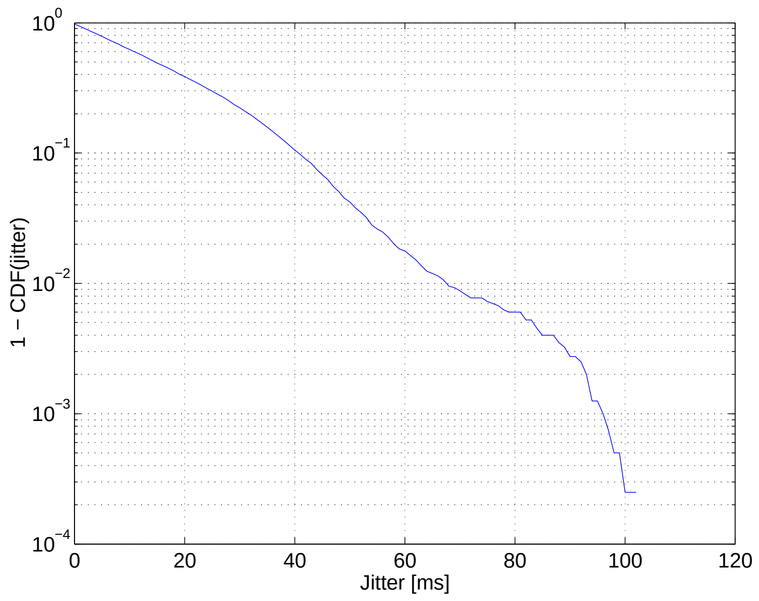

Figure 7: 1 – CDF(jitter) (average one-way delay = 300 ms, average jitter = 20 ms).

Although the emulator provides a way to set bandwidth limitations, tests have shown that the corresponding behaviour is not as expected. When such a bandwidth limit is set, NIST Net will reduce the output rate, but will buffer all the packets that cannot be sent. Once the buffer fills, it is emptied by sending all packets, which causes the output rate to exceed momentarily the input one. This is in contradiction with our expectations of a fixed output rate combined with possible packet loss if the input rate is larger than the configured output rate. Note however that for VoIP the bandwidth requirements are known and specific to every codec. Therefore we don't require the use of this feature and is enough to study the influence of jitter and packet loss only.

The main problem we encountered with NIST Net is that introducing a variable delay leads to potential packet reordering, as shown in Appendix 1. Playing the packets out-of-order has negative effects on perceived quality, since the voice samples are not played in the order in which they were produced; therefore an additional reordering step in mandatory. This operation can be done based on the timestamps included in the Real Time Protocol (RTP) header.

Since our VoIP application (see next section) produces a data file containing all received packets, it is possible to reorder the packets before emulating their reception and constructing the output wave file. This has certain side effects that are discussed in Appendix 1. Therefore we decided to reorder packets directly in the dejittering buffer, as explained in the Section 2.3.2.

3.3 Speak Freely

The VoIP application we used for our tests is Speak Freely version 7.6a [Wil-**], a Linux freeware tool. The application sends voice data over the network using a certain encoding, and ensures decoding and playback at the receiving end.

Voice data is encoded using certain standard codecs, that compress audio data as coming from an input source (e.g. microphone). On decoding data is decompressed so that it can be played by a sound card. Codecs are necessary because in VoIP voice data must be packetised and then transformed into a network level representation. This implies sampling with a certain rate (usually 8 kHz) and a sample size (usually 1 byte). Data is either applied a basic encoding, such as A-law or µ-law, or more advanced compression methods are used. Each codec has specific characteristics regarding the compression level and the perceived quality it achieves, which are usually a trade-off between network utilisation efficiency and audio quality.

3.3.1 Features

Speak Freely implements a series of codecs that are all available when using the default built-in protocol: G.711, G.726, GSM, LPC, LPC-10, GSM + 2X, CELP. However some of them lack when using RTP or VAT protocol. This is because for RTP or VAT there are standards regarding how the encapsulated data is encoded. RTP related information is provided in [Sch-03]; this RFC defines the identifiers for a series of codecs. Thus, in order to preserve compatibility with other applications, compression modes not specified in these standards cannot be encapsulated in RTP or VAT. This is the case for the GSM + 2X, LPC-10 and CELP codecs in SpeakFreely, for which information could not be provided in the tables 2 to 4 except when using the built-in SpeakFreely protocol.

Although not available in Speak Freely, G.729 [G.729] is a codec of interest since it has a high compression rate (8 kbps voice data) with a relatively high PESQ score of about 3.5. Since this codec was not included, we obtained the reference ITU-T implementation for the version G.729AB of the codec, which seems to be the most widely used. We integrated it into the post-processing stage of our experiments. Tests with this codec are performed using an already compressed speech file at the input of the VoIP application and adjusting the packet size according to the the level of compression of G.729. Only RTP protocol was used, therefore in G.729 case data for the other protocols is not presented in tables 2 to 4.

In our first experiments we tried to use the output of the PC speakers directly when assessing the UPQ for VoIP by recording it to a wave file. Due to the very poor quality of the recording (caused by low quality of the signal paths, which include analogue-to-digital and digital-to-analogue conversions) this proved to be impossible. The solution we found was to emulate in software the actions taken after packet reception by the VoIP application, that is mainly dejittering. Therefore Speak Freely has been modified to write a file that allows us to create an output wave file equivalent to the voice signal the VoIP application would produce on PC speakers. This additional functionality enables us to compare different ways of dealing with VoIP packets at the receiver (e.g. dejittering strategies) using exactly the same network traffic.

The changes are as follows. For each VoIP packet that is received the modified version of Speak Freely writes into an output file the uncompressed voice data along with its timestamp (the time at which it is supposed to be played) and its arrival time at the receiver PC[*]. The file obtained is the input of a program we wrote, "process_voice_data", that emulates voice playback and the dejittering buffer as described in Section 2.3.2.

* The timestamp is obtained from the RTP header. The arrival time is determined on the receiving PC using the function `gettimeofday`, a standard C library function that has a precision in the order of milliseconds. This precision is sufficient given that packets contain at least 40 ms of voice data.

3.3.2 Evaluation

Speak Freely capabilities are summarised in the tables below. They were determined based on the application documentation, the source code, as well as several tests. We denoted the Speak Freely communication protocol by SFP. All data rates are given for simplex connections. They should be doubled for normal duplex operation, when the end nodes communicate bidirectionally.

Table 2 shows the transmission rates (VoIP data rate, network rate and packet rate), as well as network packet sizes, for the various codecs when using RTP as a transport protocol. Note that the codecs used in our full-scale experiments are presented in the first part of the tables.

Codec	***Data rate [kbps]***	***Packet size [bytes]***	***Network rate [kbps]***	***Packet rate [pps]***
G.711	64	378	75.6	25
G.726	32	382	38.2	12.5
GSM	13	190	19	12.5
G.729	8	170	17	12.5
LPC	5.3	116	10.6	11.4
LPC-10	2.4	*N.A.*	*N.A.*	*N.A.*
GSM + 2X	6.6	*N.A.*	*N.A.*	*N.A.*
CELP	4.8	*N.A.*	*N.A.*	*N.A.*

Table 2: Transmission rates and packet sizes for the codecs available with RTP (packet overhead = 58 bytes).

As it can be seen from Table 2 each codec has different characteristics regarding the data rate and the packet rate it produces. These are correlated to the compression level each codec has. See Chapter 4 for more details about each studied codec.

Tables 3 and 4 below compare the available transport protocols and all codecs that can be used with the corresponding protocols.

Codec	***SFP data [bytes]***	***RTP data [bytes]***	***VAT data [bytes]***	***SFP audio [samples]***	***RTP audio [samples]***	***VAT audio [samples]***
G.711	488	320	320	488	320	320
G.726	489	324	324	972	640	640
GSM	332	132	132	1600	640	640
G.729	*N.A.*	112	*N.A.*	*N.A.*	640	*N.A.*
LPC	142	58	58	1600	640	640
LPC-10	70	*N.A.*	*N.A.*	1800	*N.A.*	*N.A.*
GSM + 2X	488	*N.A.*	*N.A.*	3196	*N.A.*	*N.A.*
CELP	146	*N.A.*	*N.A.*	1920	*N.A.*	*N.A.*

Table 3: Raw data and audio data sizes corresponding to each codec for the available protocols.

Note for example in Table 3 that although the amount of network data sent in a packet is almost the same (e.g. for G.711 and G.726), the amount of audio data in samples varies due to the different level of compression. This is also reflected by the values shown in Table 4, where the amount of audio data in milliseconds is provided.

Compression used by voice codecs is lossy. Therefore reconstructed audio data after decompression is not identical to the input audio data that was compressed. Hence a degradation of the perceived quality appears, which increases usually with the compression level. This is illustrated by the last column of Table 4 which shows the average PESQ scores, a measure of VoIP UPQ, that are obtained when using the corresponding codecs.

Codec	***SFP audio [ms]***	***RTP audio [ms]***	***VAT audio [ms]***	***Maximum PESQ score***
G.711	61	40	40	4.3
G.726	122	80	80	3.8
GSM	200	80	80	3.4
G.729	*N.A.*	80	*N.A.*	3.5
LPC	200	80	80	1.9
LPC-10	225	*N.A.*	*N.A.*	2.5
GSM + 2X	400	*N.A.*	*N.A.*	2.7
CELP	240	*N.A.*	*N.A.*	3.2

Table 4: Audio data size (in milliseconds) and the maximum PESQ score corresponding to each codec for the available protocols.

Since RTP is more widely spread for VoIP communication, we selected it as the transfer protocol in our experiments. Based on their availability in Speak Freely and the corresponding maximum PESQ scores, the following codecs were selected for study: G.711, G.726 and GSM. In addition, G.729 was integrated and used as previously mentioned. Chapter 4 presents the results obtained, with a separate section for each studied codec followed by a comparison of the codecs. This can be used to select the codec offering the best quality under certain network conditions.

Silence detection

Silence detection is a feature present in some of the VoIP applications. Basically this works by selecting an activity threshold below which no sound is transmitted. This feature is sometimes called Voice Activity Detection (VAD). G.729AB integrates this silence detection in the codec itself. From a few tests we were able to determine that the decrease of the PESQ score when using VAD with respect to not using it was generally low, below 0.1 in all cases. The important effect is that the amount of network traffic decreases by 40 % (see Table 5).

	PESQ score	***Compressed size [bytes]***
VAD disabled	3.48	33600
VAD enabled	3.42	20224

Table 5: PESQ scores and compressed size of voice data for G.729 codec with VAD disabled or enabled ("female_all" recording).

In general silence detection is performed by the VoIP application and the other codecs do not have built-in silence detection mechanisms; therefore this form of VAD is application dependent, since each application can use its own algorithm for silence detection. In addition this feature is generally not enabled by default in VoIP applications, since users are sceptical about its effect on quality. Due to all these reasons we decided not to enable VAD during our tests for any of the codecs, given that we mainly intend to provide a baseline for their behaviour.

3.4 Preliminary tests

We ran several types of preliminary tests in order to determine the baseline conditions for running our experiments. To obtain the results presented below we used the RTP protocol, as in all our tests. Given that our application sends within each RTP packet at least 40 ms worth of uncompressed audio data (80 ms when sending compressed data), the quanta of varying the playback delay during tests is 40 ms.

The codec used for all these preliminary tests was G.711. No packet loss occurred during experiments. The artificial average delay introduced by NIST Net was of 300 ms and the average jitter was varied depending on the type of test we performed.

3.4.1 Playback delay calibration

Several tests were performed to calibrate the playback delay. These experiments were performed for traffic with an average jitter of 20 ms, which was considered as being representative for normal network conditions under moderate stress. The intended jitter was therefore set to 20 ms in NIST Net. The measured jitter at network level, computed based on the information collected by the monitoring system, was of about 20 ms as well. *PLAYBACK_DELAY* was given values from 0 to 200 ms in increments of 40 ms. A value of zero is equivalent to no dejittering buffer being present.

Experiments were performed with both female and male voice recordings (obtained along with the ITU-T P.862 standard [P.862]) as follows:

- female1, female2, female3: three different 8 second long recordings of the same female voice;
- male1, male2, male3: three different 8 second long recordings of the same male voice;
- female_all/male_all: the 24 second long recordings obtained by the concatenation of the three female/male recordings mentioned above.

As an example we provide here the transcription of the file "female_all":

> *You are the perfect hostess. Are you going to be nice to me? You know my outlook on life. I jumped at least two feet. He took out his pipe and lit up. It was the same in the public bar.*

A total of 20 tests was performed for each input file. Results marked female123 or male123 are the averages of the results obtained for female1, female2 and female3 or male1, male2 and male3, respectively.

Figures 8 and 9 show the results obtained for female recordings, average PESQ scores and their standard deviation. It is noticeable that the plots are similar for the different recordings.

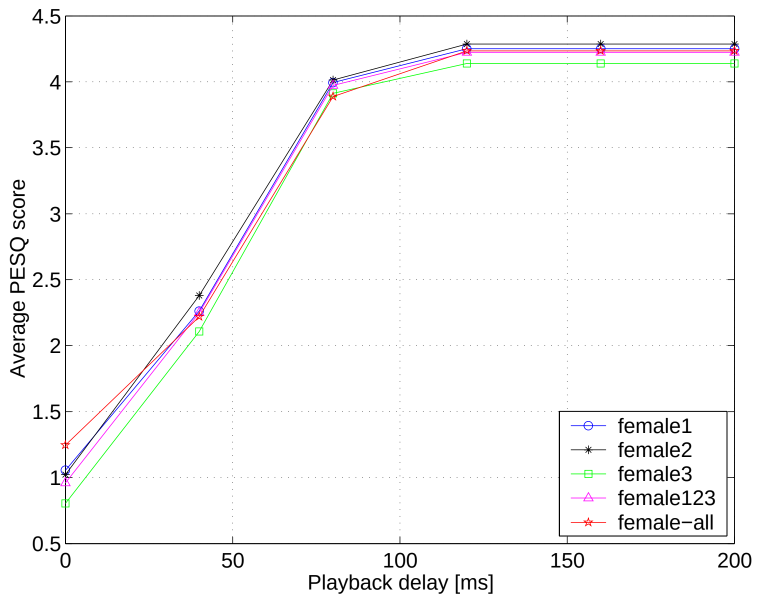

Figure 8: Average PESQ score for female voice recordings (average jitter = 20 ms).

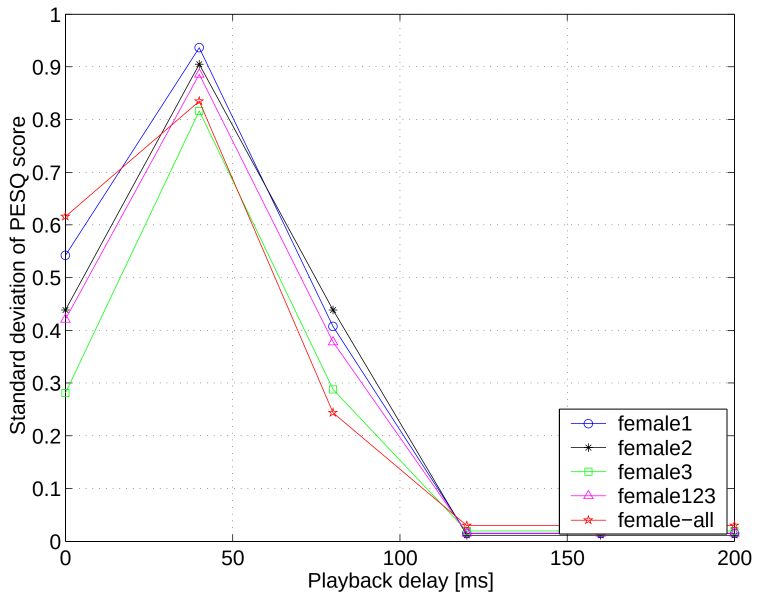

Figure 9: Standard deviation of PESQ score for female voice recordings (average jitter = 20 ms).

Figures 10 and 11 show the average PESQ scores and their standard deviation for male voice recordings. Again the plots are similar for the various voice files.

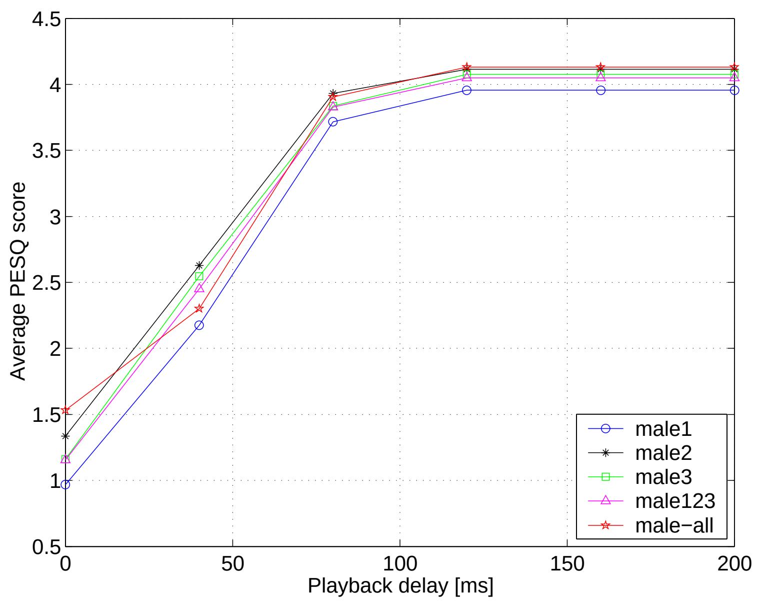

Figure 10: Average PESQ score for male voice recordings (average jitter = 20 ms).

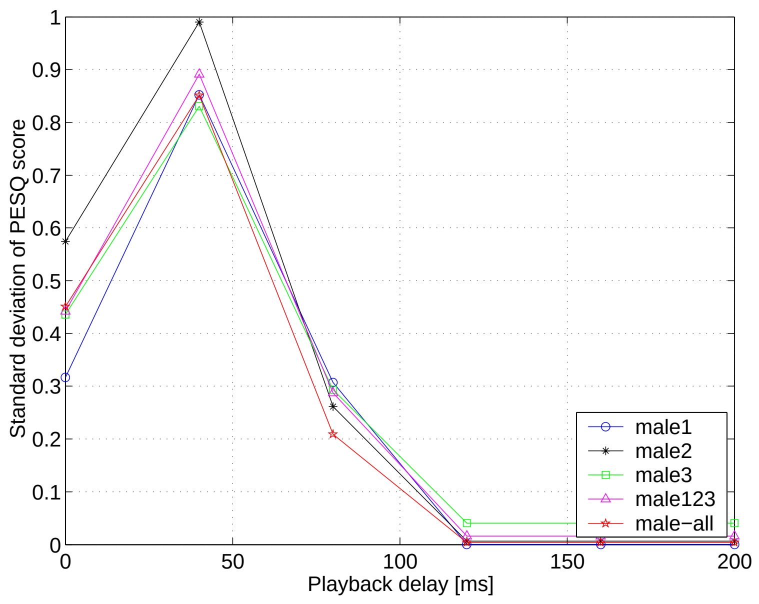

Figure 11: Standard deviation of PESQ score for male voice recordings (average jitter = 20 ms).

It is clear from the graphs displayed above that there are slight differences between the results obtained with different voice recordings depending on the speaker. As a general remark, the PESQ scores obtained for the female speaker recordings are higher by approximately 0.2 than those obtained for the male speaker recordings.

Comparing the results we obtained for a 24 s recording to those computed by averaging the results for three 8 s original recordings we conclude there is no significant consistent difference in the average PESQ score. On the other hand the standard deviation of the results obtained using "female_all" and "male_all" is generally lower, by up to 0.2, probably due to the larger number of packets involved. Therefore we decided to continue performing our tests using the "female_all" file.

As far as the dejittering buffer is concerned, we note that for a playback delay of 80 ms, which is equivalent to two VoIP packets when encoding with G.711, the PESQ score obtained for “female_all” is quite close to the maximum (4.0 instead of 4.25 – cf. Figure 8) but not equal to it. This is explained by the fact that, even if the average jitter is only 20 ms, there are instantaneous jitter values larger than 80 ms (cf. Figure 7) that prevent this playback delay from fully countering the effect of jitter.

The contribution of a playback delay of 80 ms to the overall end-to-end delay still allows for it to be below the interactivity thresholds (i.e. below 150 ms) if network delay doesn't exceed 70 ms. Therefore for all the experiments presented in Chapter 4 *PLAYBACK_DELAY* had a value of 80 ms.

3.4.2 Playback delay and jitter relationship

For these tests the voice recording was “female_all”. We wanted to study the influence of the playback delay on quality when network jitter varies. *PLAYBACK_DELAY* was given values from 0 to 200 ms in increments of 40 ms*. A value of zero is equivalent to no dejittering buffer being present. Network jitter varied between 0 and 40 ms in steps of 10 ms.

The plot in Figure 12 shows that perceived quality increases with playback delay, since the dejittering buffer compensates for the negative effects of jitter. In the same time, comparing the graphs obtained for different values of the jitter one infers that the value of the playback delay that insures optimum quality depends on the level of jitter.

* Given the fact that the maximum one-way delay that insures good interactivity is 150 ms [17, 12], it was not necessary to study larger playback delays.

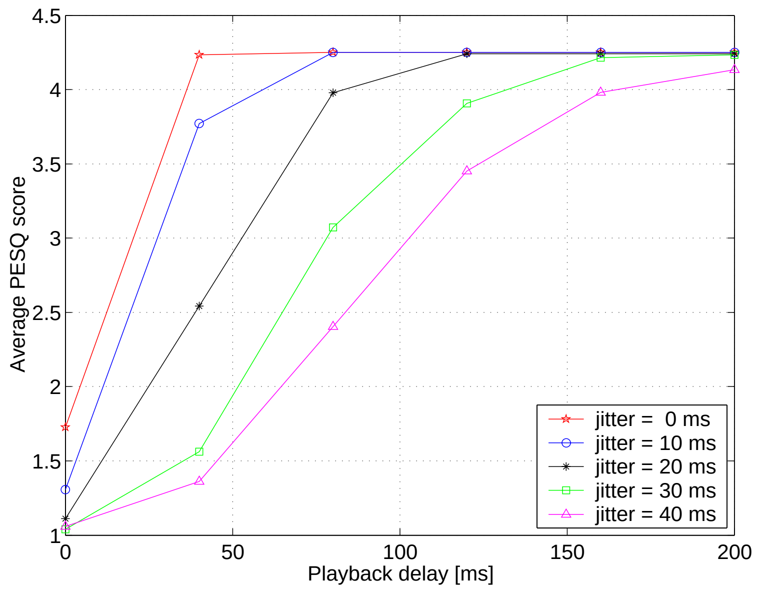

Figure 12: Average PESQ score vs. playback delay for different values of the average jitter.

Note that an intended jitter of 0 ms means effectively a very low measured network jitter, lower than 0.02 ms in all cases. However even this low jitter affects strongly quality if the playback delay is zero. As it can be seen from Figure 12, the line corresponding to a jitter of 0 ms and a *PLAYBACK_DELAY* of 0 ms, the PESQ score is only around 1.7. This is because if a packet is not on time when it has to be played, then it will be dropped, hence the quality decreases significantly.

The same data but plotted in a different way shows the dependency of the PESQ scores on jitter. One can see that the larger the playback delay, the larger the jitter value for which quality starts decreasing.

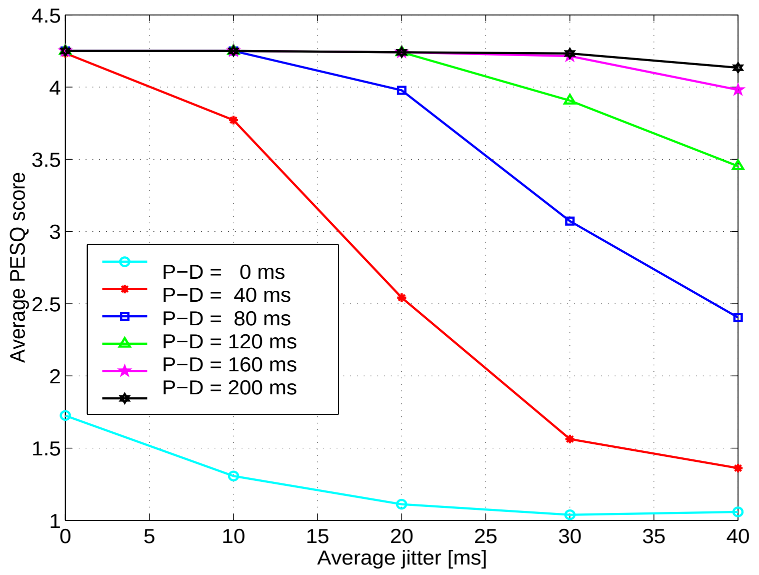

Figure 13: Average PESQ score vs. average jitter for different values of the playback delay (denoted by P-D).

3.4.3 Dejittering loss

We quantified the dejittering loss (see Section 2.3.2), in order to provide a better insight in the way in which jitter influences quality. The plot in Figure 14 shows the dependency of the dejittering loss (i.e. jitter-induced loss) on jitter for all the studied codecs. We used the same conditions as those for all the experiments, i.e. *PLAYBACK_DELAY* had a value of 80 ms and the average delay was 300 ms. Packet loss at network level was zero.

For the moment the influence of the codec on the dependency between dejittering loss and the average jitter is not very clear, but examining this issue in more depth was not considered to be within the scope of our research. We assume there may be an influence of the number of packets and the amount of voice data they represent. For G.711 there is a double number of packets compared to the other codecs, since each packet represents only 40 ms of voice date, whereas G.726, GSM and G.729 packets contain 80 ms. This could explain why the same jitter induces a larger loss for G.711.

However we cannot explain yet why G.729 seems to perform better than the other codecs which have the same characteristics in terms of number of packets and amount of voice data content. A possibility would be that, due to the way in which tests were performed, for all codecs except G.729 decoding happened in real time and is taken into account by the arrival times are the receiver. For G.729 voice data is written compressed in the intermediate file and is only decompressed before dejittering. This didn't seem to be a problem since we have always assumed that decompression takes a relatively low and fixed amount of time, therefore it doesn't introduce any additional jitter. This might not be the case in practice however.

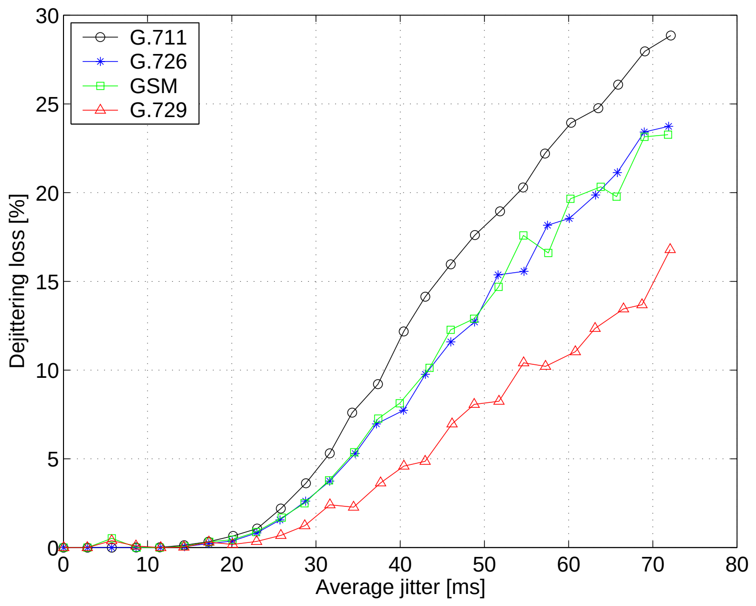

Figure 14: Dejittering loss vs. average jitter (PLAYBACK_DELAY = 80 ms).

3.4.4 Codec evaluation

When using codecs to send voice over a communication channel, due to the lossy voice data compression, there is an implicit decrease in perceived quality; hence there is a specific maximum score that can be obtained with each codec. The corresponding average scores that we obtained in our setup are given in Table 4 for the codecs available in Speak Freely. All codecs used in our tests (G.711, G.726, GSM and G.729) had an intrinsic quality above the acceptability level, while two of them (G.711 and G.726) have a quality above the PSTN level.

In Table 6 below we show the possible variations for the same codec among different test voice recordings of a female and male speaker, as supplied along with [P.862]. The codec used to obtain these results is G.711.

Voice recording	***PESQ score***	***Voice recording***	***PESQ score***
female1	4.25	male1	3.96
female2	4.29	male2	4.12
female3	4.15	male3	4.11
female123	4.23	male123	4.06
female_all	4.25	male_all	4.14

Table 6: Maximum PESQ scores obtained for different files using the G.711 codec in our experimental setup.

These tests show that there are slight differences between the results obtained with different voice recordings. As a general remark, the PESQ scores obtained for the female speaker recordings are higher by approximately 0.2 than those obtained for male speaker recordings. However, for the same speaker, the variation between the scores for various recordings is remarkably small, within 0.05 with respect to the average.

3.5 Test configuration

This section presents the settings that were used to perform all our full-scale experiments. We configured the NIST Net network emulator to introduce artificial QoS degradation. Based on our previous tests we decided to focus on packet loss rates ranging from 0% to 15% in steps of 0.5%. Intended jitter ranged from 0 ms to 75 ms in steps of 3 ms.

For the purposes of our experiments we have also set a fixed one-way delay of 300 ms. This allows for the possibility of a variation around it of up to 4 times the maximum jitter value (i.e. 75 ms) without getting negative delays. This is the maximum variation one can obtain with NIST Net – see equation (6) in Section 3.2. Note that for large values of jitter packet reordering occurs. This is dealt with by the dejittering emulation algorithm as described in Section 2.3.2, with a 80 ms playback delay.

A test consisted of 806 runs, one for each combination of packet loss rate and jitter values mentioned above. For each run we computed the actual packet loss rate and the actual average jitter (determined according to formula (5) in Section 2.3). This is done based on the traffic descriptors produced by the monitoring system. We also calculated the PESQ score, by comparing the output wave file to a reference wave file. The reference wave file is a female recording, “female_all”, selected based on preliminary tests presented in the previous section. The recording has a length of 24 seconds and is obtained by concatenating three test samples provided along with the ITU-T recommendation P.862 [P.862].

A total of 5 tests were performed for each codec in order to obtain the results presented in Chapter 4. The input voice signal for our experiments is obtained by PCM encoding the reference wave file at 64 kbps (8 kHz sampling frequency, 8 bits per sample, µ-law encoding). The input voice data was stored in a file of 192000 bytes. Sending voice data using the G.711 codec implies no additional compression; for all the other codecs this signal represents the input of the corresponding compression algorithm.

3.6 Result representation

For each codec tested we obtained 5 sets of points distributed in the loss-jitter space. Since the real values attained in each experiment for the QoS parameters differ from the intended values supplied to NIST Net, the points are not situated precisely on a grid and additional steps are required in order to be able to represent the data. There are two issues related to the representation of the results from our tests that will be discussed in this section: surface plotting and surface modelling.

Note that on the jitter axis we will always represent the average network jitter value, although dejittering will compensate for a certain amount of network jitter (below 20 ms as it can be seen from our preliminary tests). In order to identify the effects jitter has one can use the metric *dejittering loss,* as defined in Section 2.3.2.

3.6.1 Surface plotting

The raw data we obtain from our experiments are points in the space of packet loss, jitter and PESQ score. These points can be represented as a surface in the 3D space, having packet loss and jitter on the *x* and *y* axes, and PESQ score on the *z* axis. In order to build this surface which contains all test points an operation called triangulation is necessary. This stands for determining all the non-intersecting triangles that have as ends of their edges the test points. The surface is then the union of all these triangles and can be represented in a 3D space. A standard way to perform this task is to use Delaunay triangulation – we make use of the Matlab function `delaunay` for this purpose.

The Matlab function `trisurf` was necessary to plot the data in the representation described above. It shows the results of 5 tests obtained for the voice file "female_all" using the codec G.711, an average delay of 300 ms and a playback delay of 80 ms (see Figure 15). The same data is used for the plots in Chapter 4, the section dedicated to the codec G.711.

The representation described above is the most basic one and shows the raw data. We observe there is noise in the PESQ results, the surface obtained is not smooth as expected. This is probably due to the different influence losing a packet containing voice or silence has on perceived quality. We conclude based on Figure 15 that a smoothing algorithm must be used before representing the data. Several alternatives are presented next; note that they achieve smoothing on a rectangular grid, therefore surface representation doesn't require triangulation anymore.

The first method we tried is to interpolate based on the available points according to a uniformly distributed grid in the studied ranges and represent the interpolated results. Both interpolation and representation are done using standard Matlab functions: `griddata` (with cubic interpolation) and `surf`. The graph is displayed in Figure 16.

The second method we used is the following: we slide an overlapping window of fixed size in the loss-jitter space and average the PESQ scores for all points within this window. The result is assigned to the point located in the centre of the window. The surface is then plotted using these computed points. This alternative is represented in Figure 17. The optimum values for the window size were determined by comparing the graphs obtained using different values of the windows size so that to produce a relatively smooth surface while maintaining a sufficiently high level of detail. These values are 4.5 ms on jitter axis and 0.75% on loss axis; sliding is done with steps of half these values, i.e. 2.25 ms on jitter axis and 0.375% on loss axis.

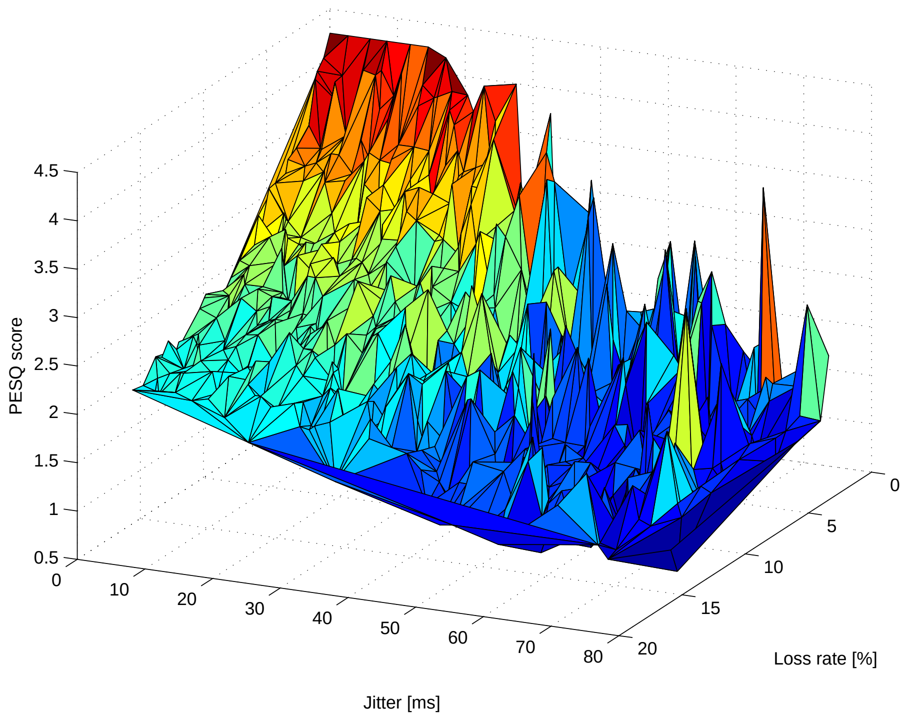

Figure 15: Raw data representation (by means of Delaunay triangulation).

Note the irregular shape of the surface represented in Figure 15. A smoothing technique is required as previously explained. Interpolation based on raw data (see Figure 16) doesn't produce satisfactory results because of the noisiness of the data to be interpolated.

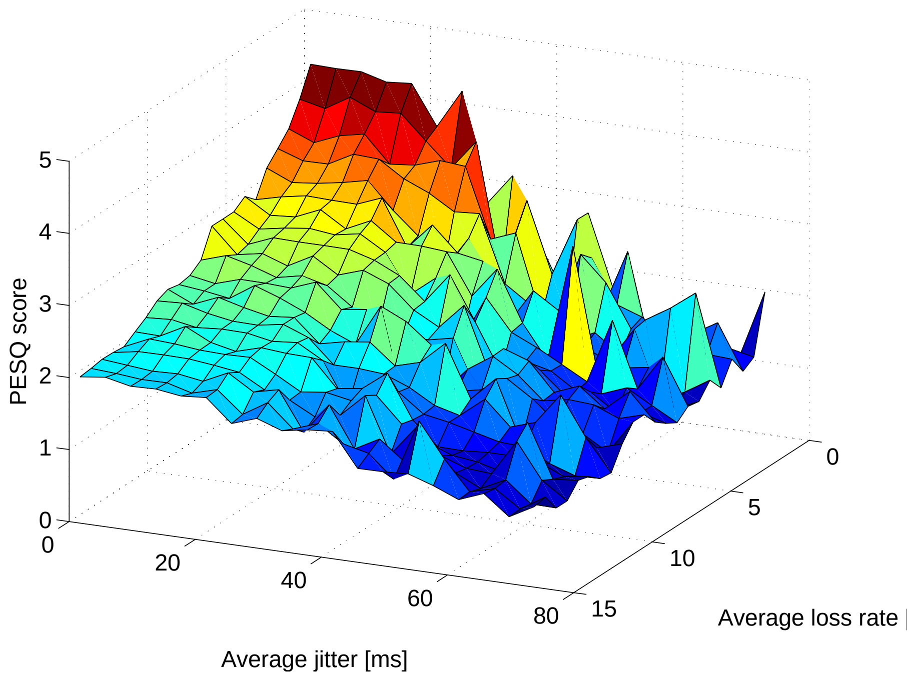

Figure 16: Representation using interpolation based on the raw data.

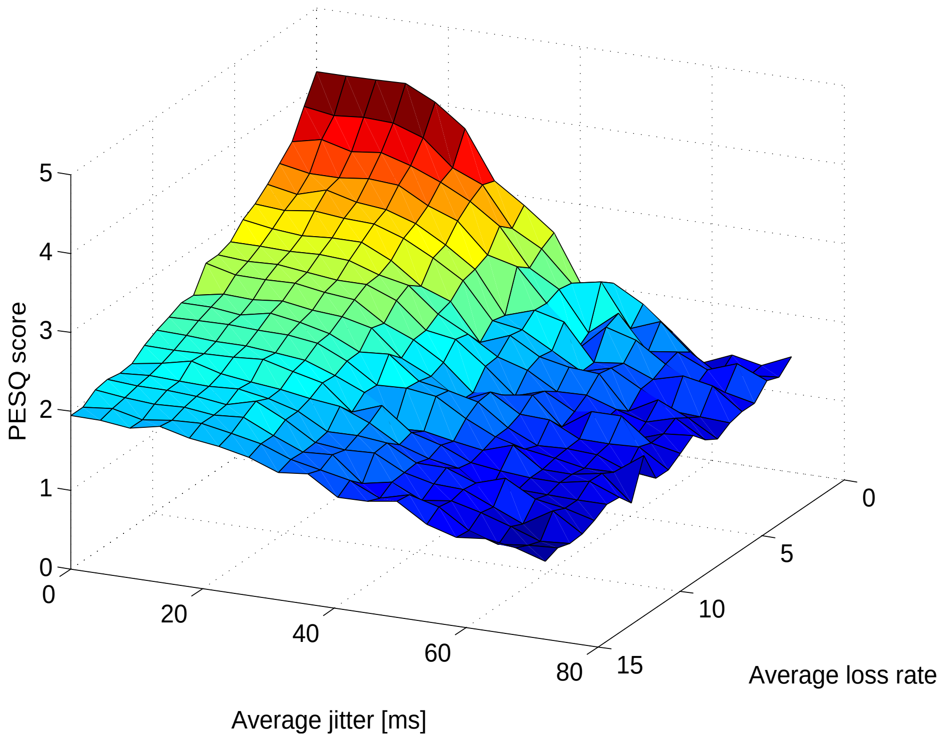

Figure 17: Representation using sliding window averaging of raw data.

Another way of averaging is to compute the mean PESQ score for the 5 points obtained in each test for any combination of packet loss rate and jitter. This average is then assigned to the intended packet loss and jitter co-ordinates and is further used for plotting the surface in Figure 18.

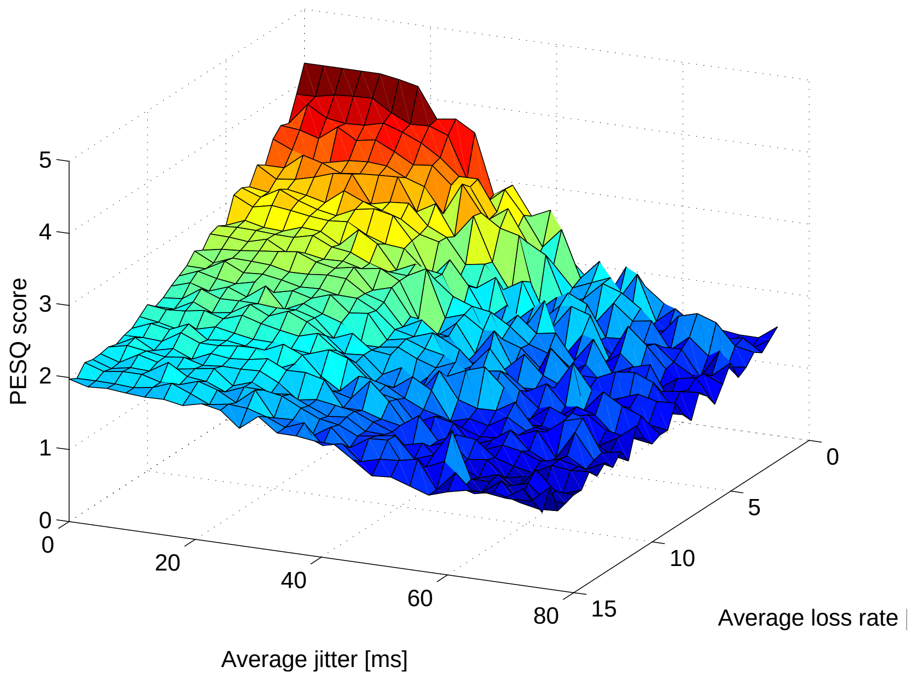

Figure 18: Representation using direct averaging of the raw data.

It is clear that all these three methods of representation of the results produce similar shape plots, with a better quality for the ones obtained using the last two methods based on averaging. From the four representation methods discussed we chose to use the third method due to the following considerations:

- raw data representation (method 1) is too noisy to provide sufficient information;
- interpolation (method 2) doesn't provide very good results if the points represent noisy data and are not spread uniformly in the studied space;
- averaging the results obtained from the five tests and representing the result at the intended jitter and loss rate co-ordinates (method 4) is not accurate, since we know that these values may differ from the actual measured QoS parameters, and it's the real values that influence quality.

The only disadvantage of method 3 is that the sliding window produces smooth transitions and therefore hides some of the details. The values we used for the plots (window size is 4.5 ms on jitter axis and 7.5% on loss axis) are however a good trade-off in our opinion and will be used for all subsequent plots of full-scale experiment results in Chapter 4.

3.6.2 Surface modelling

A mathematical description for the PESQ surface would be useful in order to reduce the amount of information that is stored in order to represent this surface. The main advantage, however, is that it would allow gaining an insight in the dependency between network conditions and PESQ scores. At a first approximation this could be a linear relationship, very valuable for producing rough estimates. Since it was not a vital issue, developing it was considered not to be within the scope of our research.

4 Codec Testing

The four codecs we performed experiments with are G.711, G.726, GSM and G.729. A separate section is dedicated to each of them. The chapter ends with a comparative analysis of the four codecs.

4.1 G.711 codec

The G.711 codec [G.711] sends data at 8 kHz with 8 bits per sample, resulting in a data rate of 64 kbps. The sound is in PCM format, encoded using the µ-law.

A number of 227410 bytes were transmitted. Data packets had 378 bytes each, out of which 320 bytes were voice data. This corresponds to 320 samples, which is equivalent to 40 ms of voice signal (one sample is 12.5 µs long). The packet rate was of 25 pps. Since raw voice data is 192000 bytes (see Section 3.5), the overhead was of 35410 bytes, i.e. 15.6% of the transmitted bytes. Using this codec we have obtained the results shown in Figures 19 to 22.

The data from all the tests was used to compute the average PESQ scores. Their dependency on both jitter and loss rate is displayed in Figure 19 as a 3D plot.

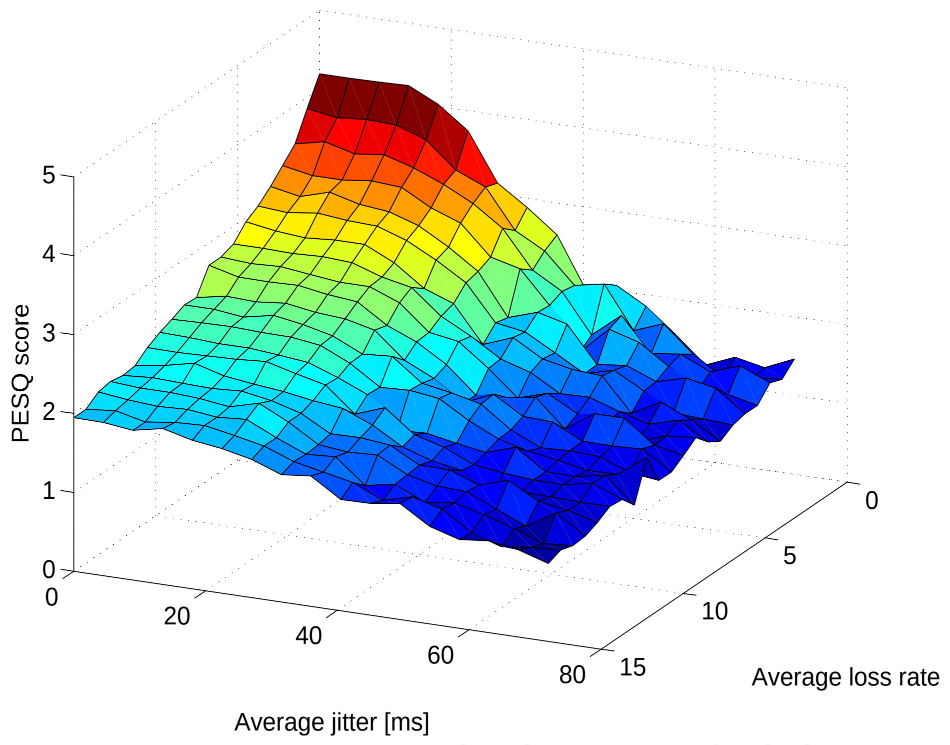

Figure 19: Average PESQ score dependency on jitter and packet loss.

Figure 20 shows the variation of the PESQ score with packet loss rate for a few values of the jitter (0, 25, 50 and 75 ms). This graph represents vertical cross-sections of the surface in Figure 19. One can establish that for loss rates up to 4% the influence is noticeable, but quality remains “good” for low jitter values; quality becomes “low” for higher loss rates, up to 15%. If jitter is high then loss doesn't change significantly the perceived quality which moreover is “unacceptable”. This is due to the high level of induced dejittering loss.

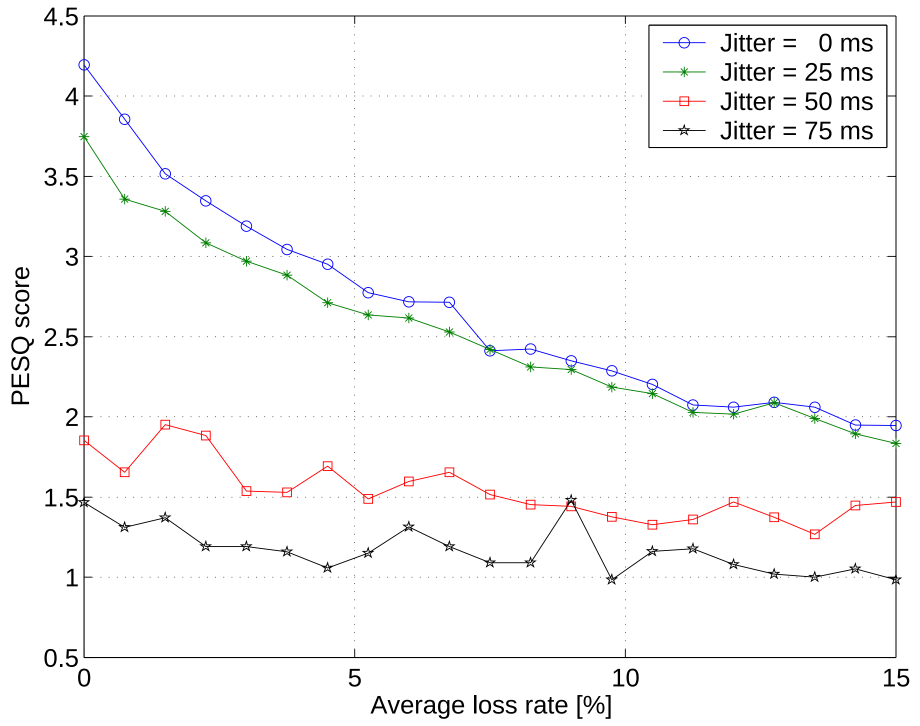

Figure 20: PESQ score dependency on packet loss rate.

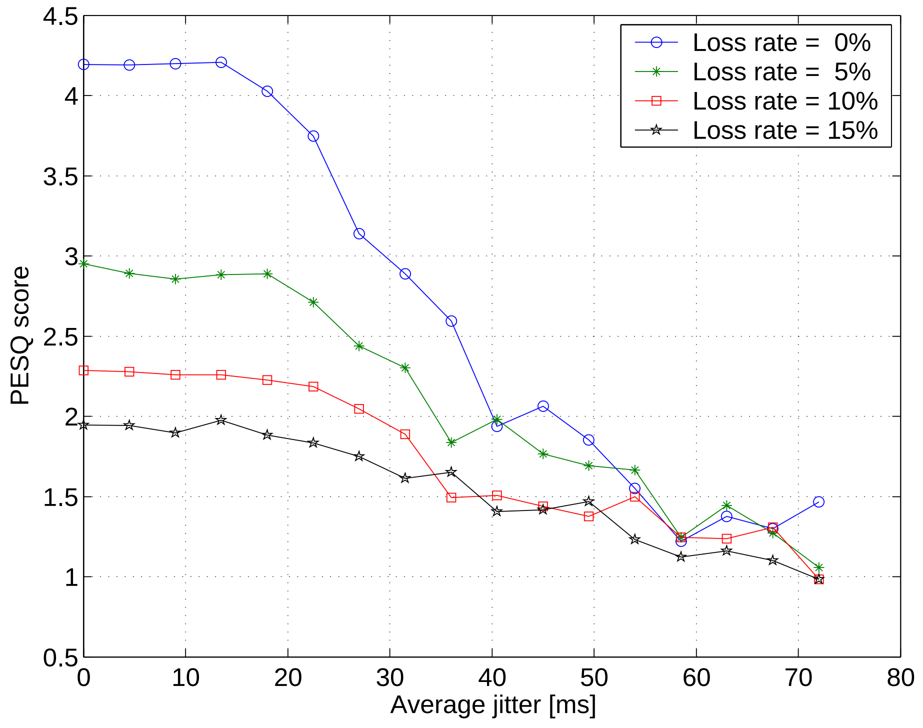

Figure 21: PESQ score dependency on jitter.

Figure 21 shows the variation of the PESQ score with jitter for a few values of the loss rate (0, 5, 10 and 15%) as vertical cross-sections of the surface in Figure 19. This figure shows that for jitter values below 20 ms there is almost no change in the PESQ scores – this is the expected effect of a playback delay of 80 ms (cf. Figure 7). Higher jitter values have a more important effect on perceived quality, the amplitude of which decreases with the increase of packet loss.

In order to have a better insight in the location of the limits of good-to-low and low-to-unacceptable quality, we made also horizontal cross-sections of the surface in Figure 19 at the PESQ score levels corresponding to these limits (PESQ score values equal to 3 and 2). For this codec one can also see the boundary of excellent-to-good quality, at a PESQ score equal to 3.8. This is the only codec we studied for which quality can be also "excellent": the corresponding area lies roughly within 0 to 1% loss rate and 0 to 20 ms jitter.

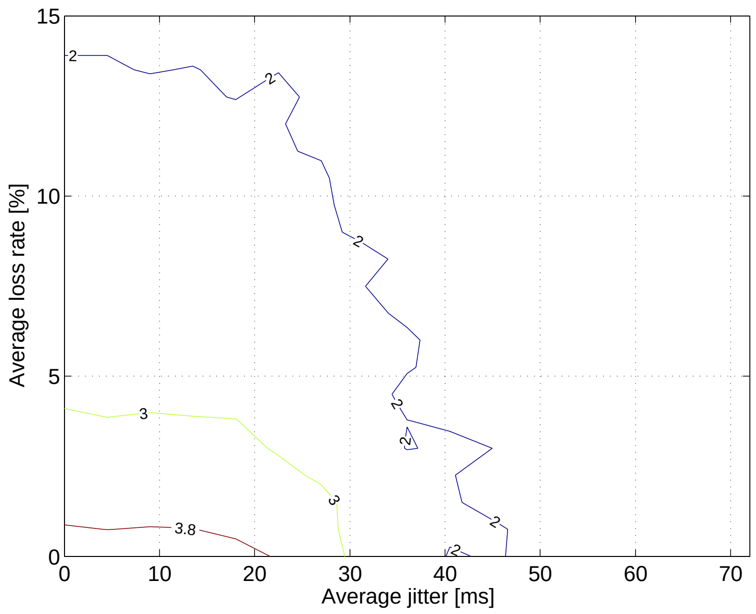

Figure 22: Contour of the boundaries between quality levels.

4.2 G.726 codec

The G.726 codec [G.726] converts a 64 kbps µ-law or A-law PCM channel to and from 40, 32, 24 or 16 kbps channel. In our application only the 32 kbps encoding is available.

A number of 115210 bytes were transmitted. Data packets had 382 bytes each, out of which 324 bytes were voice data. This corresponds to 648 samples, which is equivalent to approximately 80 ms of voice signal. The packet rate was of 12.5 pps. Since raw voice data is 192000 bytes, the compression rate is 1.67. Using this codec we have obtained the results shown in Figures 23 to 26.

Figure 23 displays the dependency of the PESQ score on both jitter and loss rate as a 3D plot. The data from all the tests has been used to compute the average PESQ scores.

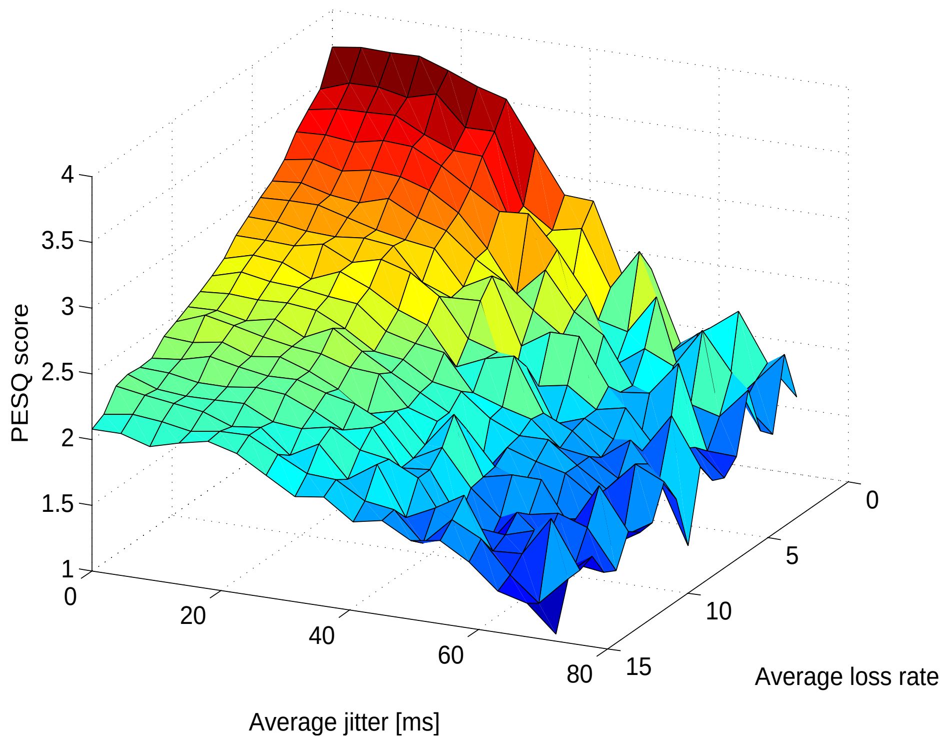

Figure 23: Average PESQ score dependency on jitter and packet loss.

Figure 24 shows the variation of the PESQ score with packet loss rate for a few values of the jitter (0, 25, 50 and 75 ms). This graph represents vertical cross-sections of the surface in Figure 23. The results are very similar to those obtained for G.711 and the same limits can be established regarding the influence of packet loss on quality. One difference is the starting point of the plot lines, which is lower because of the higher compression level. At high jitter values the graphs don't look very smooth, especially for loss rates below 5%.

Figure 25 shows the variation of the PESQ score with jitter for a few values of the loss rate (0, 5, 10 and 15%). This graph represents vertical cross-sections of the surface in Figure 23. As long as jitter is below 20 ms PESQ scores are almost constant. Thereafter quality decreases and becomes “low” or “unacceptable” for jitter exceeding 40 ms, depending on the loss rate.

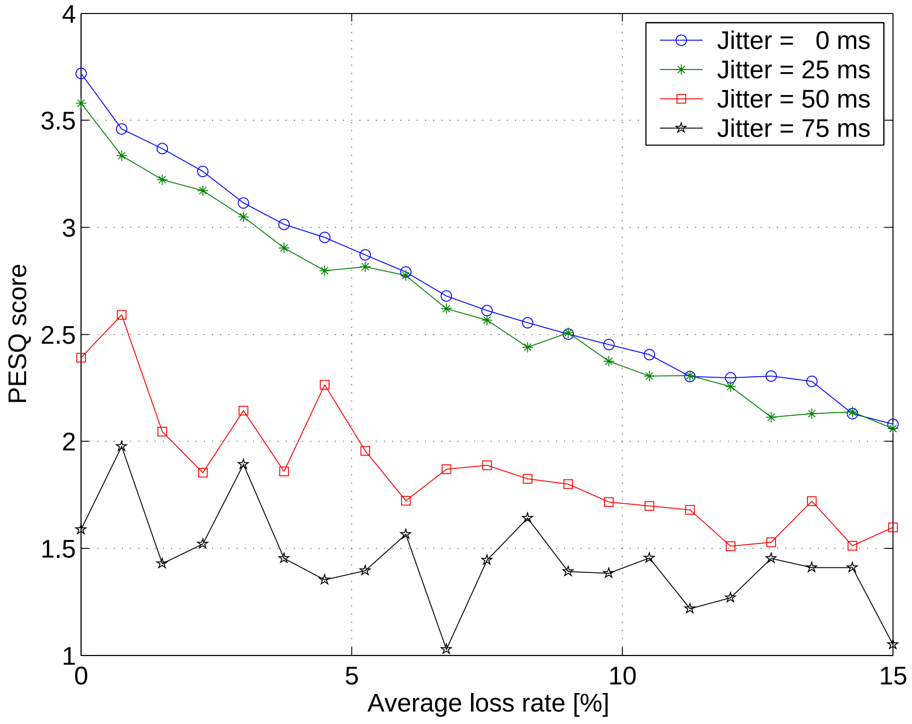

Figure 24: PESQ score dependency on packet loss rate.

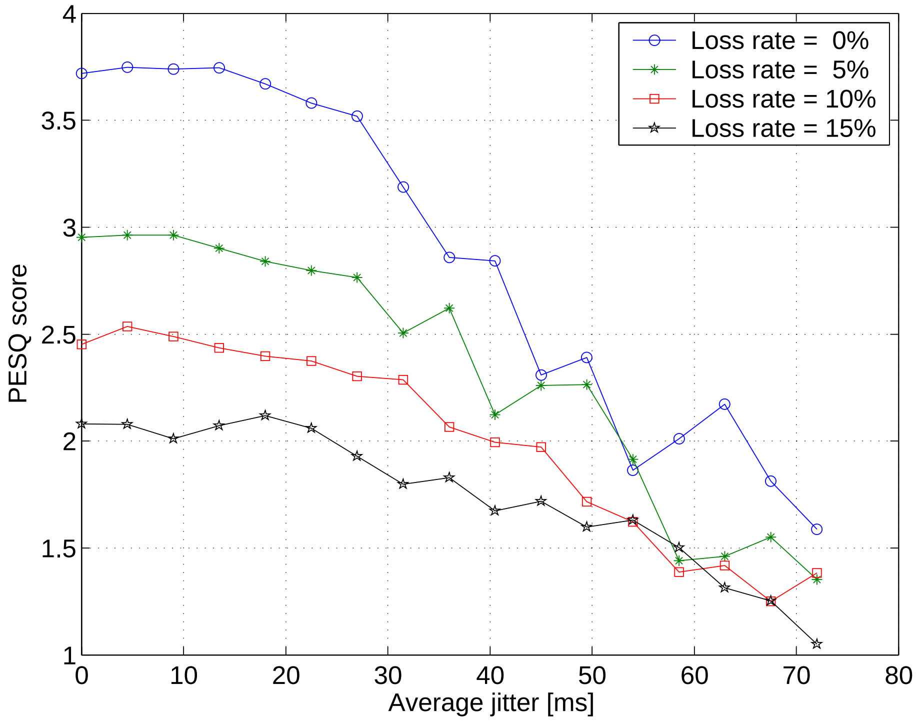

Figure 25: PESQ score dependency on jitter.

Horizontal cross-sections of the surface in Figure 23 provide a better view on the limits of good-to-low and low-to-unacceptable quality (see Figure 26). This codec cannot achieve “excellent” quality even in very good network conditions, therefore the corresponding area is not present. Note also that the boundaries for a PESQ score equal to 2 are not connected due to the residual noise still present after smoothing.

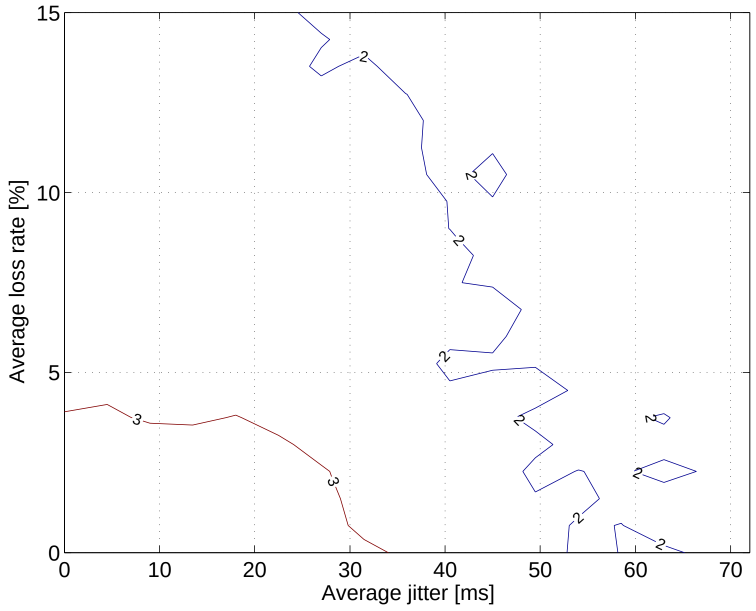

Figure 26: Contour of the boundaries between quality levels.

4.3 GSM codec

The GSM (Global System for Mobile telecommunications) codec [Rah-93] uses linear predictive coding (LPC) to compress speech data at 13 kbps.

A number of 57610 bytes were transmitted. Data packets had 190 bytes each, out of which 132 bytes were voice data. This corresponds to 649 samples, which is equivalent to approximately 80 ms of voice signal. The packet rate is 12.5 pps. Since raw voice data is 192000 bytes, the compression rate is 3.33. Using this codec we have obtained the results shown in Figures 27 to 30.

The data from all the tests has been used to compute the average PESQ scores. They are presented in Figure 27 as a 3D plots, showing the dependency on jitter and loss rate.

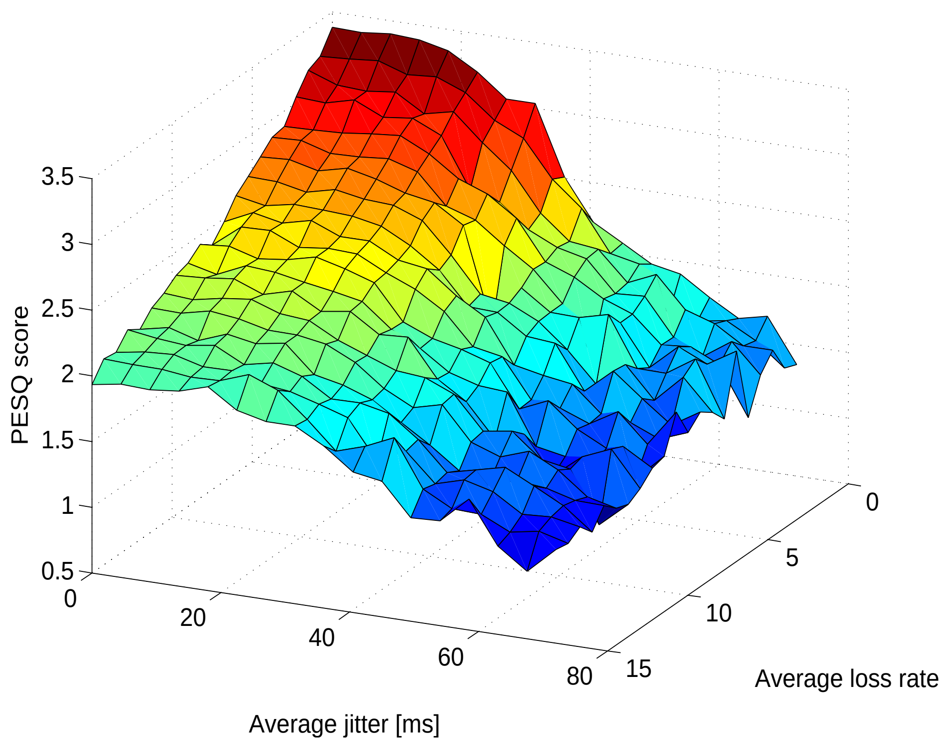

Figure 27: Average PESQ score dependency on jitter and packet loss.

Figure 28 shows the variation of the PESQ score with packet loss rate for a few values of the jitter (0, 25, 50 and 75 ms). This graph depicts vertical cross-sections of the surface in Figure 27.

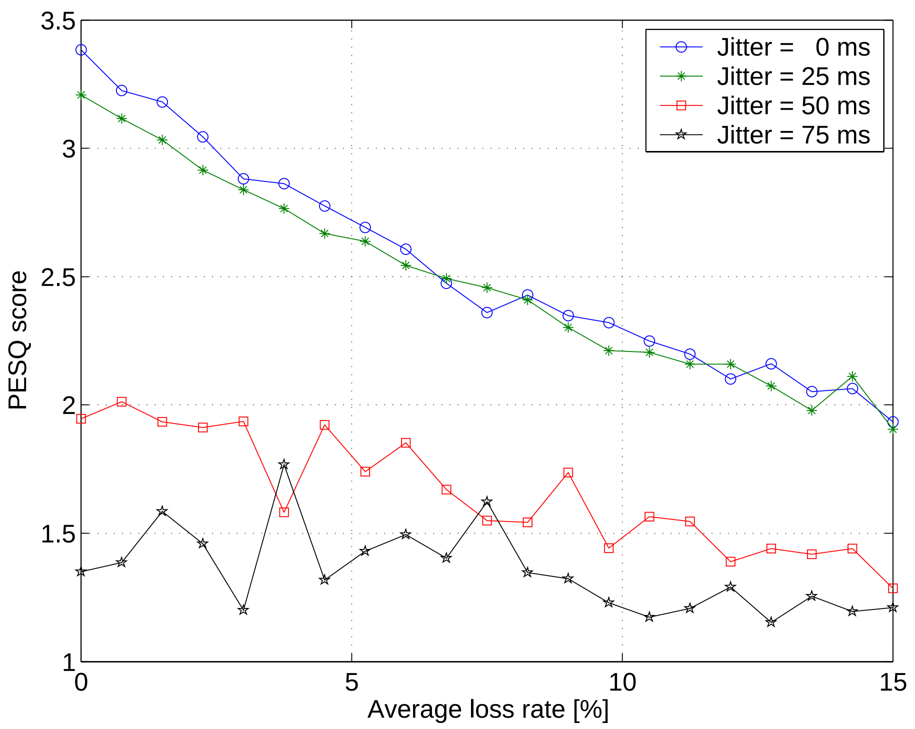

Figure 28: PESQ score dependency on packet loss rate.

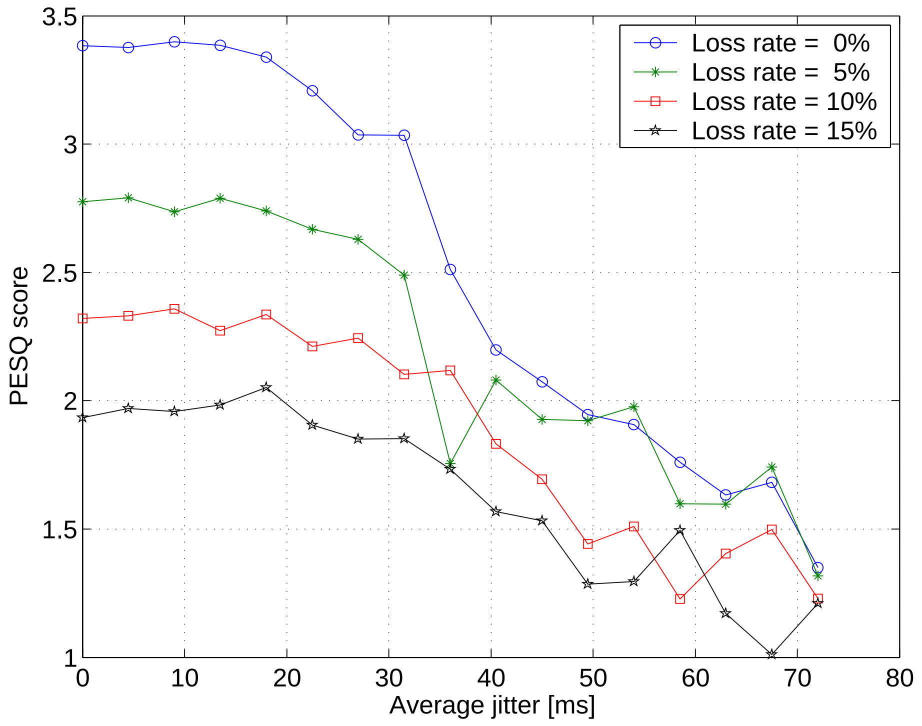

Figure 29: PESQ score dependency on jitter.

Figure 29 displays the variation of the PESQ score with jitter for a few values of the loss rate (0, 5, 10 and 15%). This graph represents cross-sections of the surface in Figure 27.

Same comments as those made for G.726 can be made regarding the previous figures. In order to better evidence the limits of good-to-low and low-to-unacceptable quality are we made horizontal cross-sections of the surface in Figure 27 that are displayed in Figure 30.

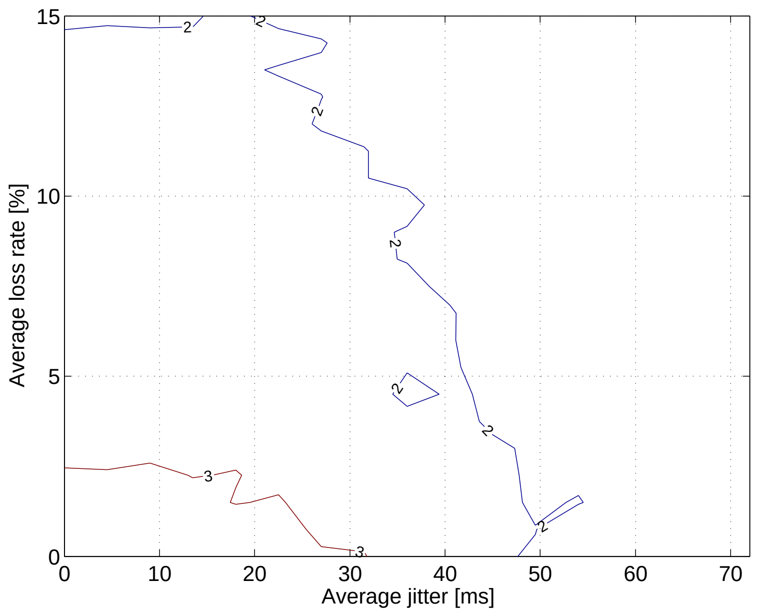

Figure 30: Contour of the boundaries between quality levels.

4.4 G.729 codec

The G.729 codec [G.729] is frequently used for VoIP communication. It sends data at 8 kbps using conjugate-structure algebraic-code-excited linear-prediction (CS-ACELP).

A number of 51610 bytes were transmitted in each test. Data packets had 170 bytes each, out of which 112 bytes were voice data. This corresponds to 640 samples, which is equivalent to 80 ms of voice signal. The packet rate was of 12.5 pps. Since raw voice data is 192000 bytes, the compression rate is 3.72. Using this codec we have obtained the results shown in Figures 31 to 34.

The data from all the tests was used to compute the average PESQ scores. They are presented in Figure 31 as a 3D plot.

Figure 32 shows the variation of the PESQ score with packet loss rate for a few values of the jitter (0, 25, 50 and 75 ms). This graph displays vertical cross-sections of the surface in Figure 31. The influence of packet loss is reduced, the decrease of PESQ score being less than 1.2 when packet loss varies from 0 to 15%.

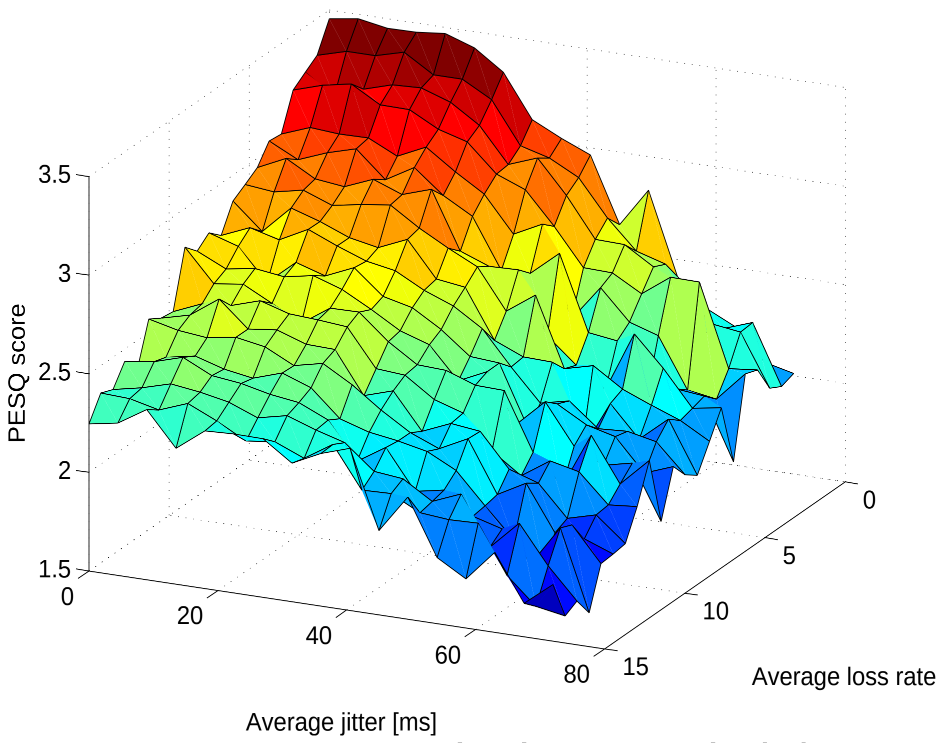

Figure 31: Average PESQ score dependency on jitter and packet loss.

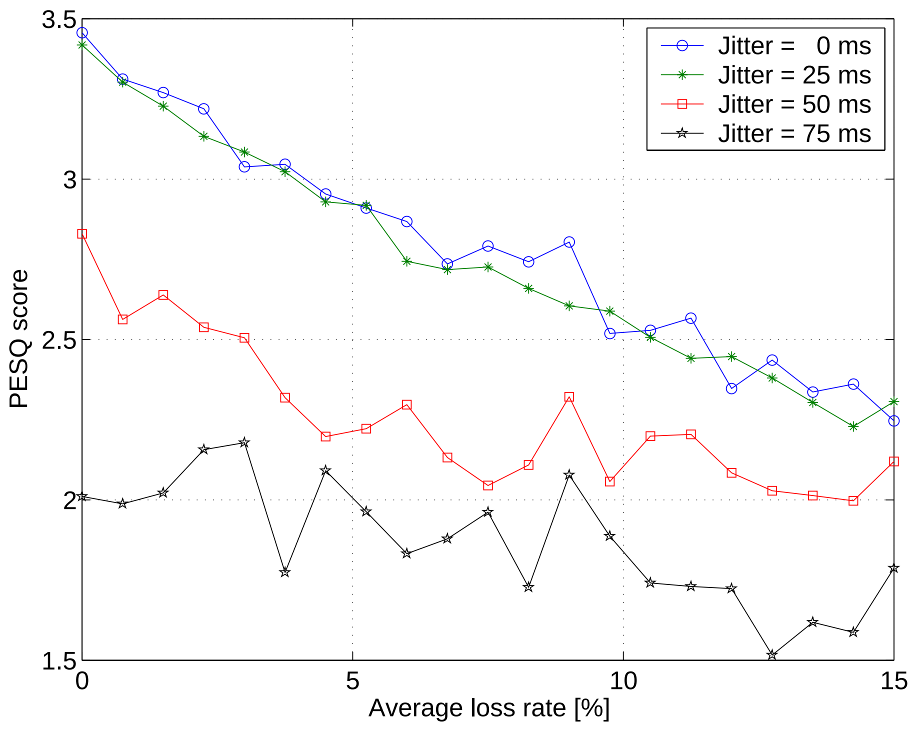

Figure 32: PESQ score dependency on packet loss rate.

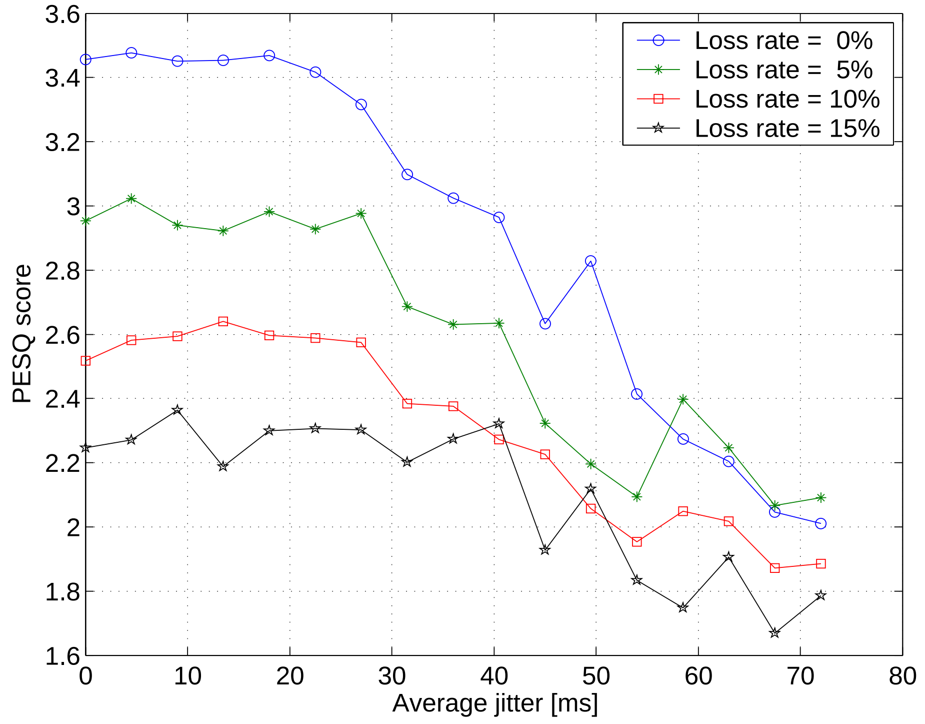

Figure 33: PESQ score dependency on jitter.

Figure 33 displays the variation of the PESQ score with jitter for a few values of the loss rate (0, 5, 10 and 15%). This graph shows vertical cross-sections of the surface in Figure 31. The influence of jitter is reduced; a decrease of PESQ score smaller than 1.5 is observed for a jitter variation from 0 to 75 ms.

To give a better insight in the position of the limits of good-to-low and low-to-unacceptable quality are we made horizontal cross-sections of the surface in Figure 31.

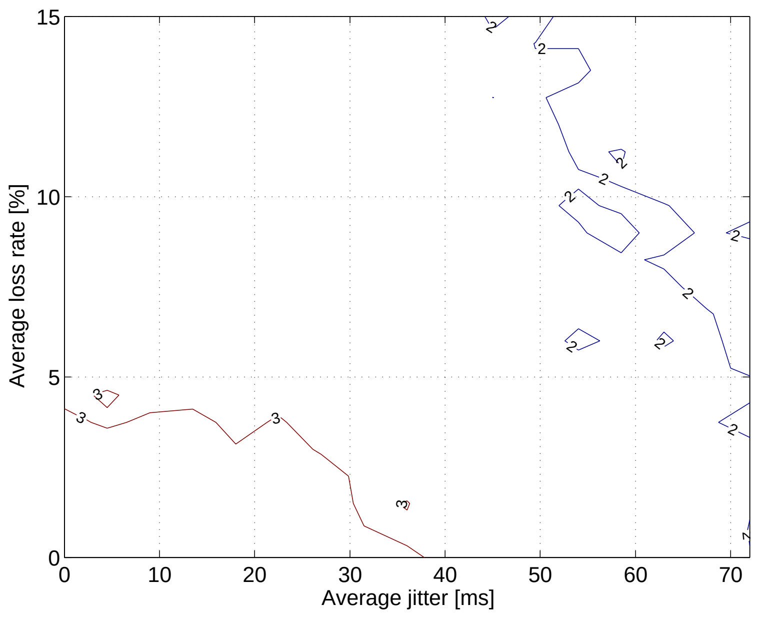

Figure 34: Contour of the boundaries between quality levels.

Note the large area of “low” acceptable quality of this codec, covering approximately 90% of the studied loss-jitter space.

4.5 Codec comparison

We present here a comparison between the codecs studied so far with plots constructed based on the 3D surfaces presented before. Each plot shows the dependency of the PESQ score with respect to one dimension while fixing a value for the other dimension. These vertical cross-sections are presented just to give an idea about the dependance of PESQ scores on loss and jitter, surfaces and contours should be used for thorough comparisons.

4.5.1 Fixed jitter analysis

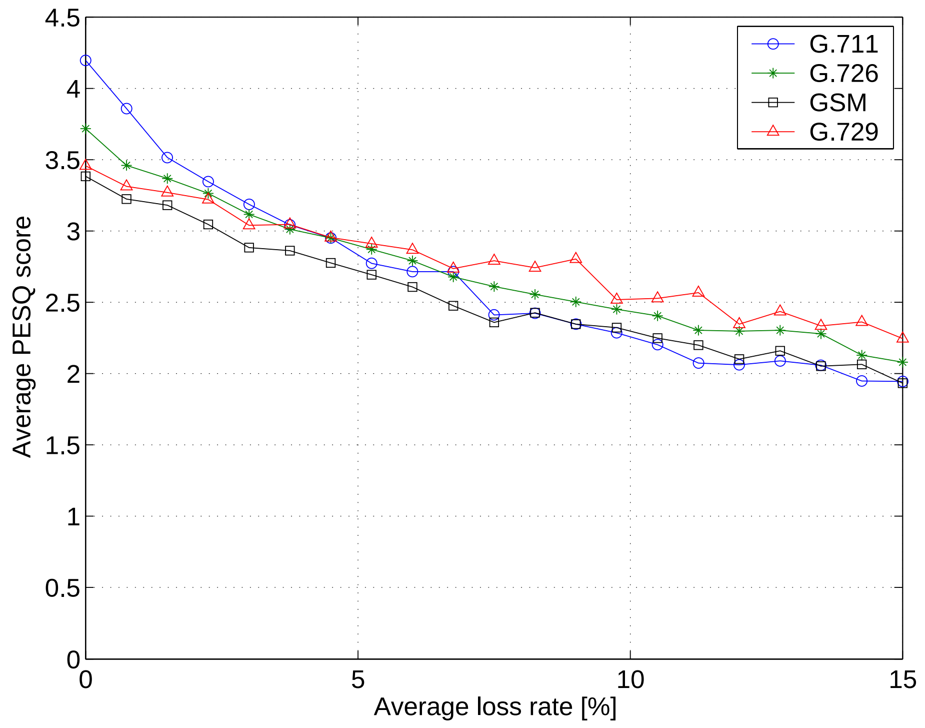

Figure 35: PESQ score dependency on packet loss rate for all codecs (jitter = 0 ms).

For a jitter of 0 ms (Figure 35) G.711 has the best performance up to a loss rate of 4%, afterwards G.729 takes over. The second best codec is always G.726. GSM has almost at all times the worst performance.

For a jitter of 25 ms (Figure 36) G.711 has the best performance only up to a loss rate of about 2%, afterwards G.729 takes over again. The second best codec is always G.726 as before. GSM has generally the worst performance, although at loss rates higher than 10% G.711 becomes even slightly worse.

When jitter is above 50 ms (Figures 37 and 38), G.729 is by far the best codec, with an average PESQ score larger than for the other codecs with up to 0.5. G.726 is again the second best, GSM is the third codec and G.711 performs the worst.

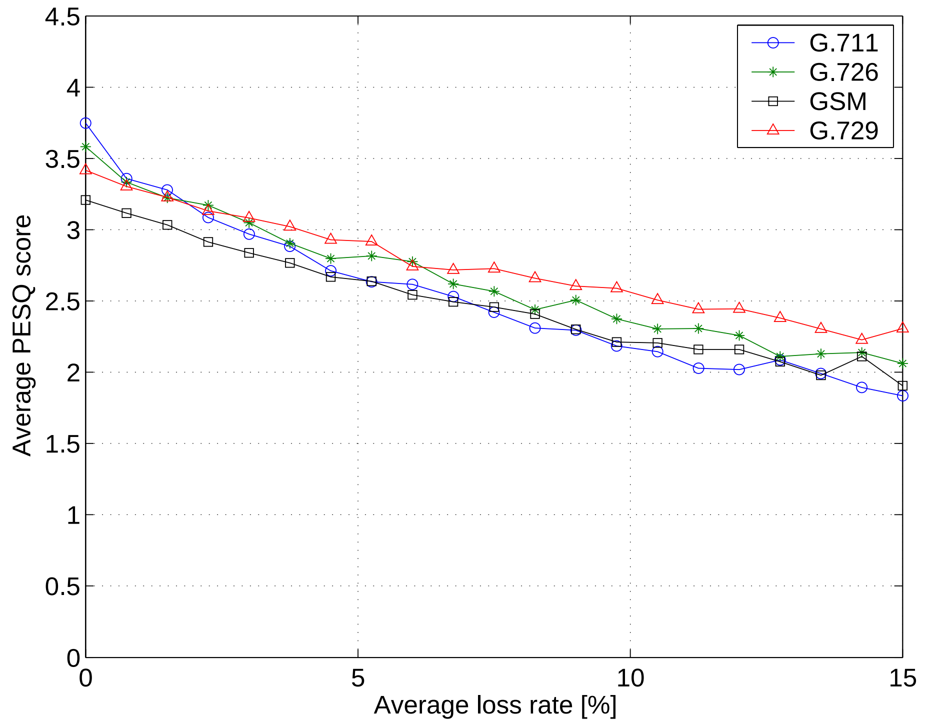

Figure 36: PESQ score dependency on packet loss rate for all codecs (jitter = 25 ms).

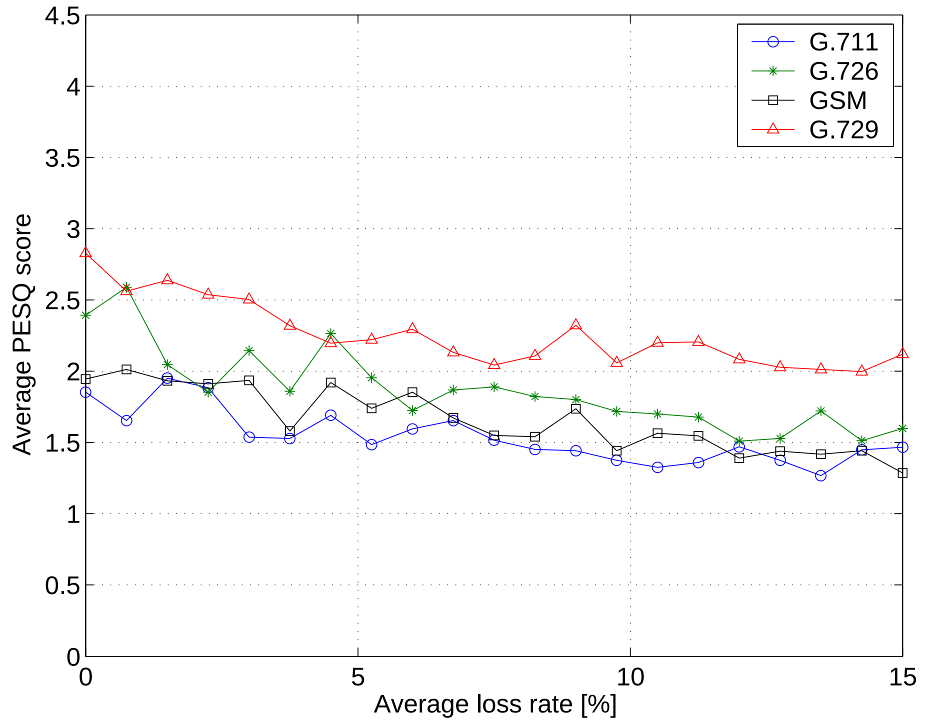

Figure 37: PESQ score dependency on packet loss rate for all codecs (jitter = 50 ms).

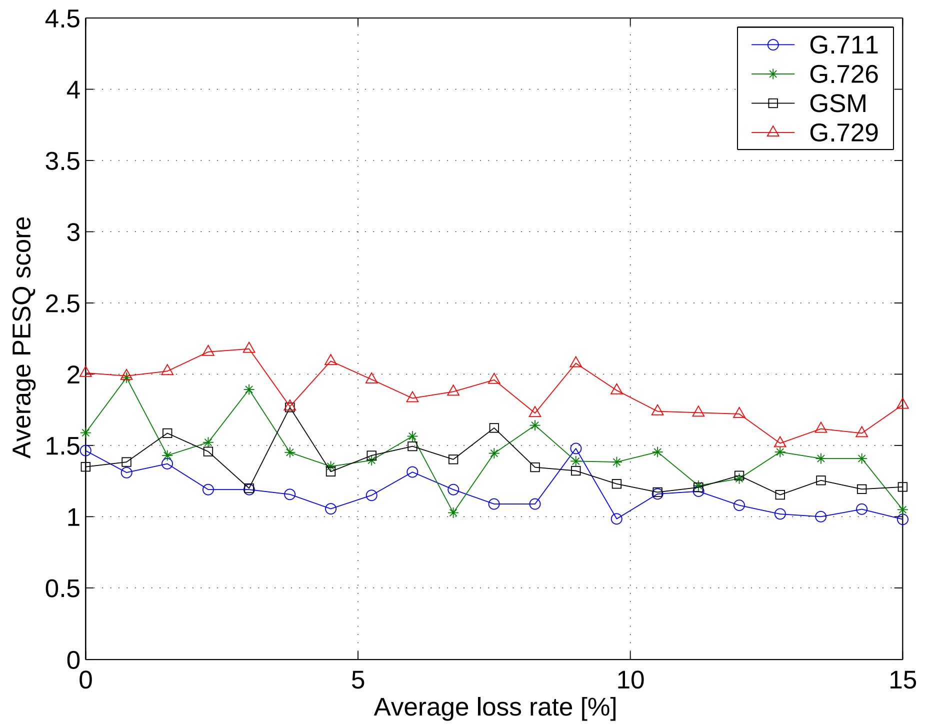

Figure 38: PESQ score dependency on packet loss rate for all codecs (jitter = 75 ms).

4.5.2 Fixed loss analysis

For a loss rate of 0% (Figure 39) G.711 performance exceeds that of the other codecs when jitter is below 25 ms. G.726 is generally the second best, but it offers the best performance within the interval 25-32 ms jitter. Starting at this point (32 ms average jitter) G.729 becomes the best codec. GSM has a rather poor performance and it's only better than G.711 for a jitter larger than 40 ms.

When the loss rate is 5% (Figure 40) all codecs have approximately the same behaviour up to a jitter of 30 ms; thereafter G.729 becomes again the best performing codec and G.711 the worst performing one.

For loss rates higher than 10% the graphs look very similar (Figures 41 and 42). G.729 provides the best perceived quality, and within the limits of “low” quality if jitter doesn't exceed 50 ms. All the other codecs have a lower performance, especially when jitter approaches 75 ms.

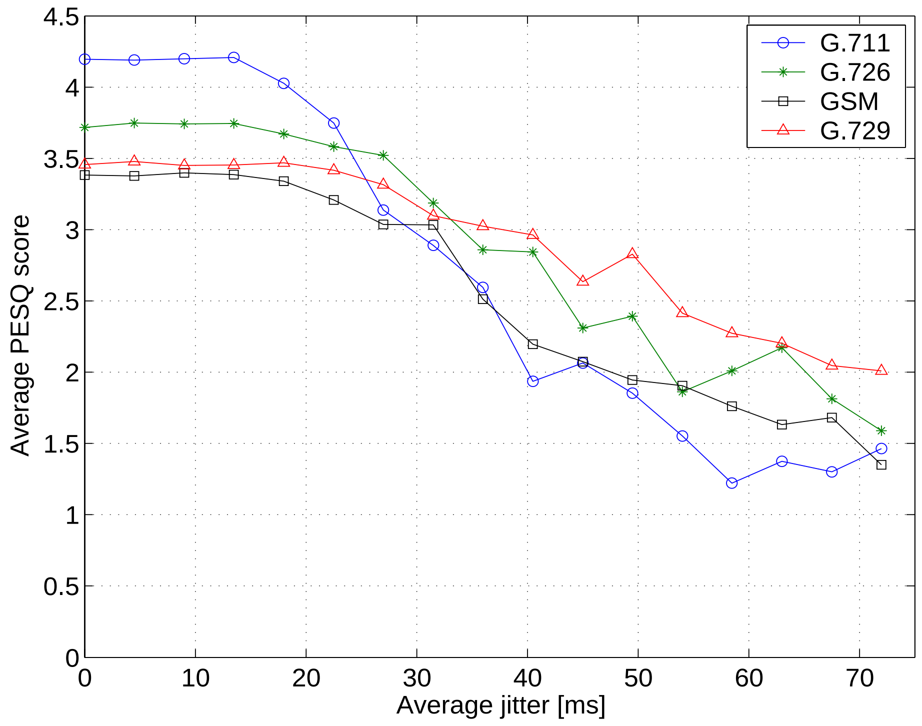

Figure 39: PESQ score dependency on jitter for all codecs (loss rate = 0 %).

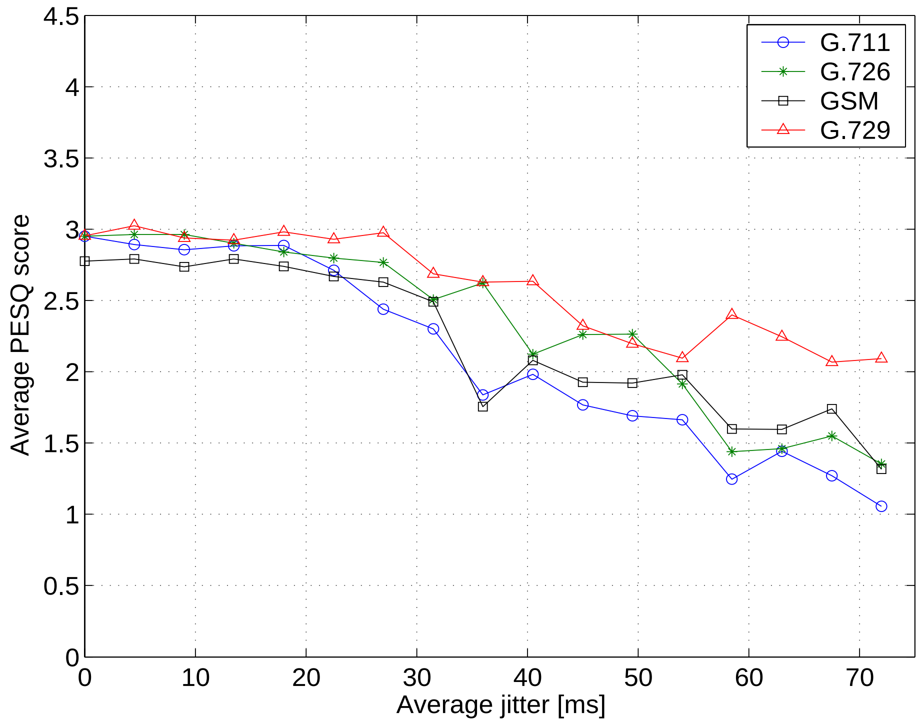

Figure 40: PESQ score dependency on jitter for all codecs (loss rate = 5 %).

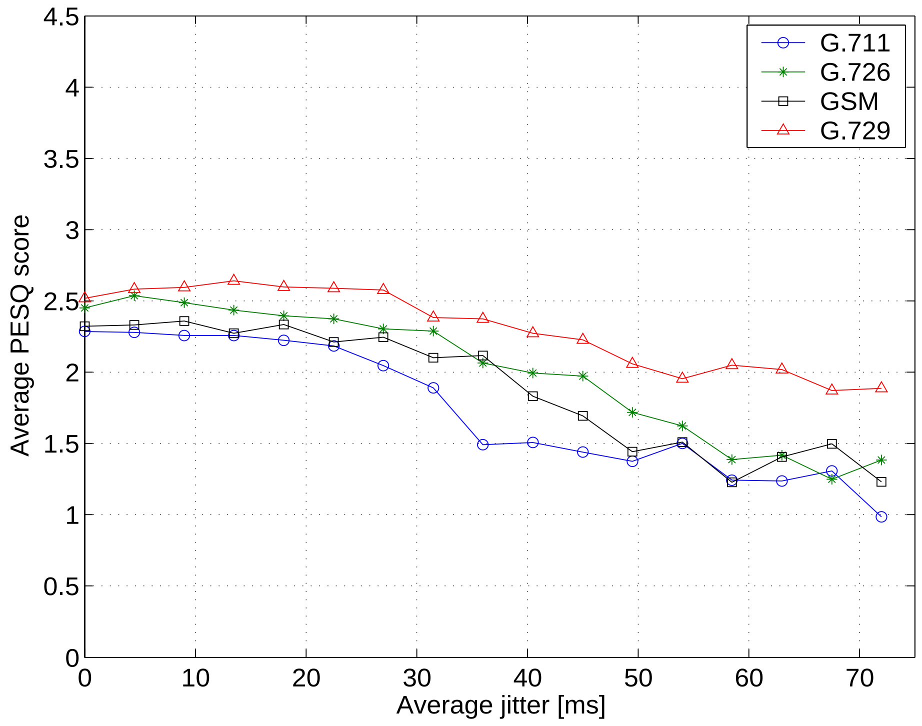

Figure 41: PESQ score dependency on jitter for all codecs (loss rate = 10 %).

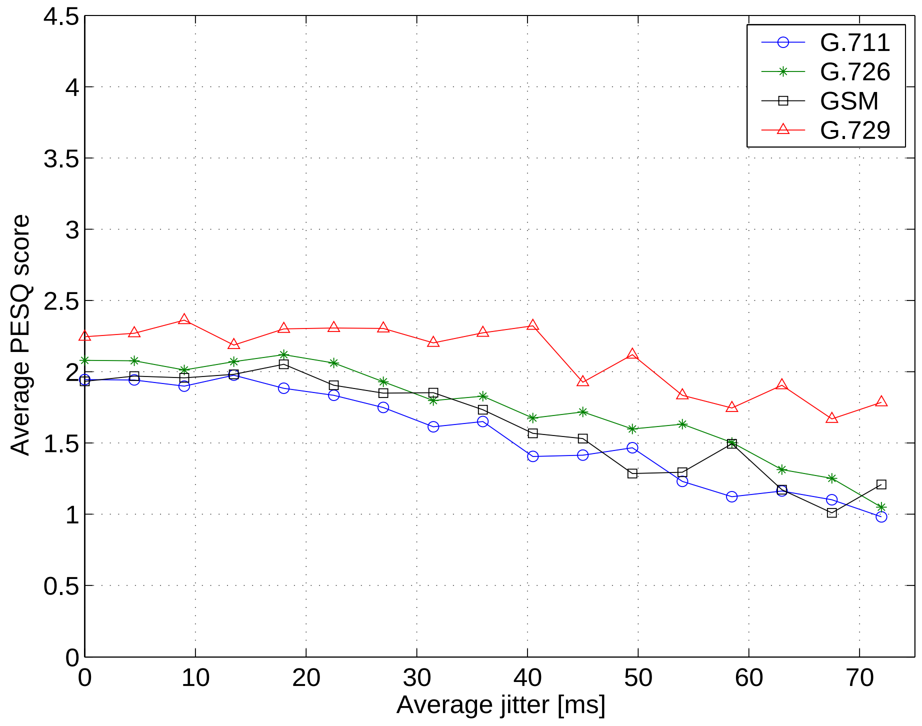

Figure 42: PESQ score dependency on jitter for all codecs (loss rate = 15 %).

4.5.3 Codec summary

To summarise the comparison we have constructed Table 7. For each codec we show the bandwidth it uses and what are the rough boundaries on QoS parameters that must be enforced in order to attain a certain quality level.

For example, G.711 provides good quality as long as loss rate is below 4% and average jitter doesn't exceed 30 ms. The same codec will provide low but acceptable quality if loss rates are roughly between 4 and 14% and jitter is between 30 and 45 ms. Outside these bounds the quality will be unacceptable.

Codec	***Good-to-low quality threshold***		***Low-to-unacceptable quality threshold***	
	Loss [%]	*Jitter [ms]*	*Loss [%]*	*Jitter [ms]*
G.711 (64 kbps)	< 4	< 30	4 – 14	30 – 45
G.726 (32 kbps)	< 4	< 35	4 – N.A.	35 – 55
GSM (13 kbps)	< 2.5	< 32	2.5 – 15	32 – 50
G.729 (8 kbps)	< 4	< 38	4 – N.A.	38 – N.A.

Table 7: Codec comparison from the point of view of quality thresholds.

Note however that for establishing exactly the limits graphs such as Figures 22, 26, 30 and 34 should be used. The table above only contains the rough limits of the rectangular surfaces that contain the areas of interest. The aforementioned figures are grouped below for an easier comparison.

A classification of the codecs can be made based on the coverage of the area corresponding to a certain quality level with respect to the area of the studied loss-jitter space. In what follows we present two such classifications, one for the area of at least *good quality* (PESQ scores larger or equal to 3) and one for the area of at least *low quality* (PESQ scores larger or equal to 2). Note however that this classification doesn't take into account the bit-rates of each codec, which are also important when making the trade-off between perceived quality and network utilization efficiency.

Codec classification based on *good quality* coverage:

1. G.729 (10.48%)
2. G.726 (9.62%)
3. G.711 (9.00%)
4. GSM (5.06%)

Codec classification based on *low quality* coverage:

1. G.729 (88.16%)
2. G.726 (60.33%)
3. GSM (51.25%)
4. G.711 (41.7%)

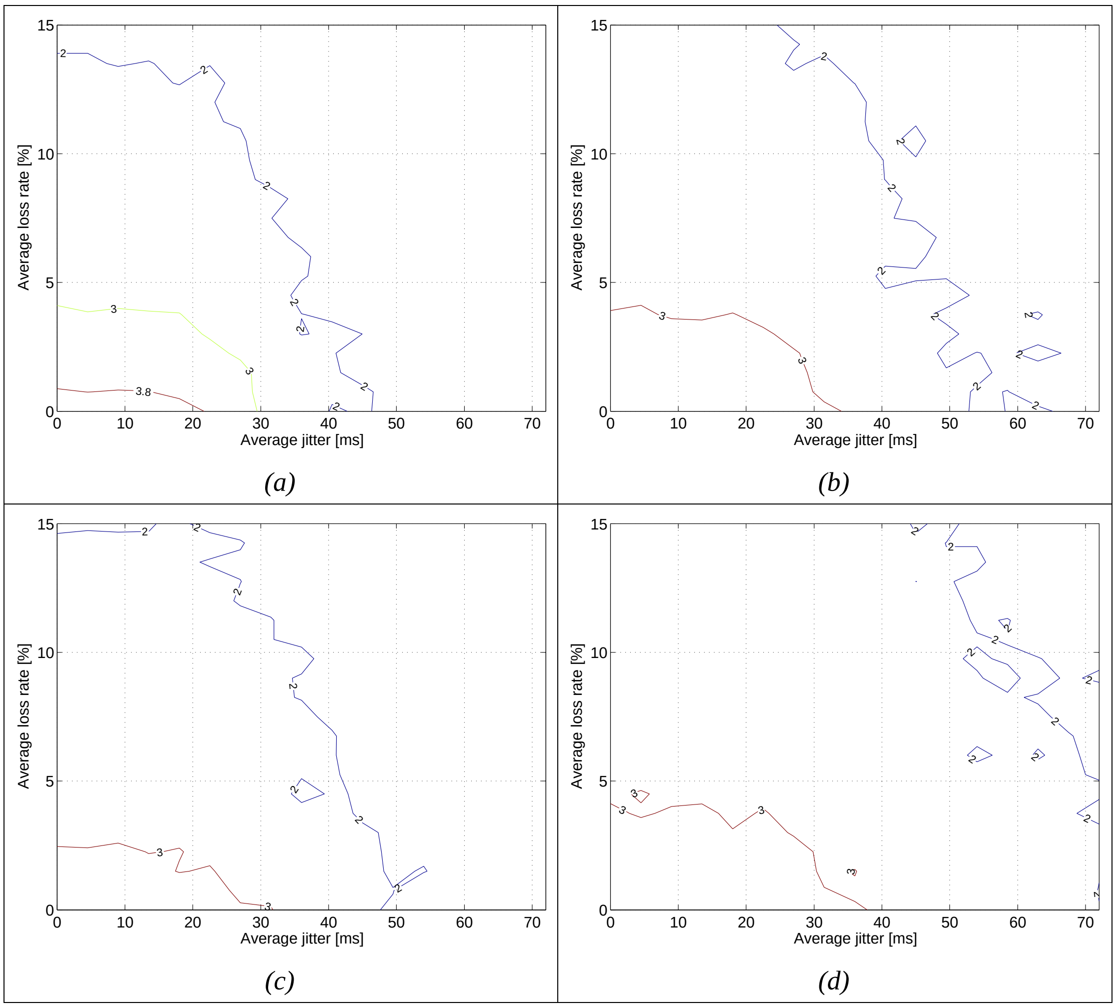

Figure 43: Contour plots of the boundaries between quality levels for the 4 studied codecs: G.711 (a), G.726 (b), GSM (c) and G.729 (d).

4.5.4 Discussions

Here are two examples of how the graphs in Figure 43 are to be interpreted and used in practice. The first one takes a regular user perspective. Regular users are not able to change the network, but they can measure the QoS level provided and adapt their applications to them. The other perspective, discussed in the second example, is that of the network provider. By means of QoS control mechanisms, such as DiffServ for example, network conditions can be mastered and tuned so that the application-level quality from a user perspective is the desired one.

The evidenced dependency of PESQ scores on jitter and loss can be used for choosing the most appropriate codec when deploying VoIP applications. For a given network, one can measure the average loss rate and the average jitter and then use the PESQ surfaces as guidelines. If the average jitter is 15 ms and the average loss rate is 0.5% then all codecs give PESQ scores higher than 3, but only G.711 produces a PESQ score corresponding to PSTN quality. If the average jitter is 15 ms and the average loss rate is 3%, then the GSM codec cannot be used if the desired PESQ scores have to be larger than 3, while the other three codecs meet this requirement. An extreme case would be the one when the average loss rate ranges from 10 to 15% and the average jitter ranges from 40 to 50 ms; in this case the only codec that can be used is G.729, which offers acceptable quality in spite of the unfavourable network conditions.

Another example can be given from a network QoS perspective. When the possibility to control network degradation exists at the service provider level or even at the LAN level, such an approach can be used to ensure a certain network QoS level is provided and implicitly a certain UPQ level is attained for an application. In the case of VoIP communication our results can be used to decided which are the boundaries that must be maintained for jitter and loss rate in order to have good perceived quality. For example, if the codec used is G.729 then these bounds are 35 ms for the average jitter and 4% for the loss rate.

5 Conclusions

The work we carried out emphasises the dependency that exists between network conditions and perceptual audio quality of VoIP communication in an objective manner. Chapter 4 should be referred to for details about each codec performance, especially Section 4.5 for a comparative analysis of the studied codecs.

A general conclusion is that the codec G.711 performs better than the other codecs as long as network conditions are good (loss rate smaller than 3% and jitter below 20 ms).

The codec G.726 gives better results than the GSM codec in the entire range of loss rates and jitter that we have studied, at the expense of a transmission rate which is 2.5 times larger. However their behaviours under the influences of loss and jitter are similar.

G.729 seems to be the most robust codec in the range of network conditions under study. It provides almost the same perceived quality (always within the bounds of "low" quality) throughout almost 90% of the loss-jitter space. A decrease only slightly higher than 1.5 is observed from zero loss, zero jitter conditions to a loss of 15% and a jitter of 75 ms, which is considerably better than what all the other codecs offer.

During our research we encountered a set of issues that were considered of smaller importance and hence could not receive the necessary attention that would lead to their solving within the timescale of our project. It is possible that in the future we shall consider some of them in more detail. A list of such issues follows:

- study the potential improvement following the use of more advanced dejittering algorithms and methods to dynamically compute the playback delay;
- study the potential improvement owing to the use of packet loss concealment techniques;
- investigate the loss pattern for the artificial packet loss introduced by NIST Net, based on the definitions in [Koo-02], since this will provide a better insight in the effects loss pattern has on VoIP quality;
- find a way to model the PESQ surfaces that would ease the understanding of the VoIP perceived quality dependancy on network conditions.

Despite this we consider that our work provides a baseline for further research on VoIP UPQ and we are confident that we have tackled all the fundamental concerns related to this task.

Acknowledgements

We would like to acknowledge the valuable discussions we had with Gerhard Heide, Chris Vowden and Annie Snelson. The suggestions of the U4EA team were useful guidelines for our research.

The support of Bob Dobinson and Neil Davies, both during our work and in the reviewing phase of this document is inestimable.

We also wish to thank our professor Vasile Buzuloiu for his prized feedback.

Brian Martin, Catalin Meirosu and Jamie Lokier had a significant contribution to the early development of the clock synchronisation system and pioneered Alteon NIC re-programming that are used in our QoS monitoring system.

Glossary

Codec (Coder/Decoder)

A codec is an entity that is responsible for encoding and decoding data representation formats. Codecs can be implemented in hardware or software. A typical example are the codecs used by VoIP applications, such as G.711, G.729 etc.

Codec also refers to compression/decompression, a technique used to reduce the size of files as they are transmitted and then expanded to normal size at the receiving point. Hence codecs are generally used to improve transmission speeds.

Constant Bit Rate (CBR)

Information that is represented in digital form by a constant stream of bits is said to have a Constant Bit Rate. Traditional formats for encoding digital voice generate CBR traffic. This is achieved by the sender transmitting packets at a fixed rate.

Cumulative Distribution Function (CDF)

This function describes a statistical distribution. It has the value, at each possible outcome, of the probability of receiving that outcome or a lower one.

Dejittering

Dejittering is the action by which some or all of the negative effects of jitter are countered in real-time applications, such as VoIP. This operation makes use of a buffer and delays playback so that packets can be played at the receiver at the same rate they had when they were sent.

Dejittering loss

When performing dejittering, such as for VoIP applications, all incoming packets are initially placed in the dejittering buffer regardless of their arrival and playback times. At playback, however, those packets for which the playback time has expired are discarded. This is called dejittering loss and is the direct way in which jitter affects voice quality.

Global System for Mobile telecommunications (GSM)

Global System for Mobile telecommunications, a pan-European mobile telephone system operating in the 900 MHz frequency band, based on digital transmission and cellular network architecture with roaming. GSM is the standard accepted in most of Europe, the Middle East, Africa, Australia and Asia.

International Telecommunication Union – Telecommunication division (ITU-T)

An agency of the United Nations, headquartered in Geneva that furthers the development of telecommunication services world-wide and oversees global allocation of spectrum for future uses.

Internet Engineering Task Force (IETF)

A body who make technical and other contributions to the engineering and evolution of the Internet and its technologies. It is the main group engaged in the development of new Internet standard specifications.

Internet telephony

See Voice over IP.

Linear Predictive Coding (LPC)

A method of digitally encoding analogue signals that uses a single-level or multi-level sampling system in which the value of the signal at each sample time is predicted to be a linear function of the past values of the quantized signal.

Mean Opinion Score (MOS)

In voice communications, particularly for VoIP applications, the Mean Opinion Score provides a numerical measure of the subjective quality of human speech at the destination end of the circuit. The scheme uses subjective tests (opinionated scores) that are mathematically averaged to obtain a quantitative indicator of the system.

Packet Loss Concealment (PLC)

An algorithm used in codecs to conceal lost or dropped packets. The algorithm may do this by playing the last packet received again or voice samples constructed based on the last received packets. The goal is to render packet loss unnoticeable to the listener, but this is done at the expense of computational resources and additional delay.

Perceptual Evaluation of Speech Quality (PESQ)

PESQ is an objective method for predicting the subjective quality of narrow-band telephony and speech codecs. PESQ has been thoroughly tested and has demonstrated acceptable accuracy for the following applications: codec evaluation and selection, live network testing using digital or analogue connection to the network, testing of emulated and prototype networks.

Perceptual Speech Quality Measure (PSQM)

This test provides numeric values of approximate speech intelligibility over a communication channel. The metric includes effects such as noise, coding errors, packet reordering, phase jitter and bit error.

Playback delay

The amount of time playback is delayed by when performing dejittering in network applications, such as VoIP. It can be either fixed or variable (the latter is used with the aim of maximising the perceived quality).

Public Switched Telephone Network (PSTN)

The regular old-fashioned telephone system that is accessed by telephones, private branch exchange trunks, and data arrangements. Completion of the circuit between the call originator and call receiver in a PSTN requires network signalling in the form of dial pulses or multi-frequency tones. The circuit remains allocated for the entire communication duration.

Pulse Code Modulation (PCM)

Pulse Code Modulation is a digital scheme for transmitting analogue data. The signals in PCM are binary, that is there are only two possible states. This is true no matter how complex the analog waveform happens to be. Using PCM, it is possible to digitize all forms of analog data, including full-motion video, voices and music.

Quality of Service (QoS)

Quality of Service is a measure of network performance that reflects the network's transmission quality and service availability. QoS may be quantitatively indicated by channel or system performance parameters, such as signal-to-noise ratio, bit error ratio, delay, message throughput rate and call blocking probability.

QoS may also mean a collective measure of the level of service delivered to the customer. In this case QoS can be characterised by several basic performance criteria, including availability, error performance, response time and throughput, lost calls or transmissions due to network congestion, connection set-up time, and speed of fault detection and correction. Service providers may guarantee a particular level of QoS (defined by a service level agreement) to their subscribers.

Real-time Transport Protocol (RTP)

The Real-time Transport Protocol is an Internet protocol standard that specifies a way for programs to manage the real-time transmission of multimedia data over either unicast or multicast network services. RTP was designed by the IETF's Audio-Video Transport Working Group to support video conferences, VoIP applications etc. RTP does not in itself guarantee real-time delivery of multimedia data, since this is dependent on network characteristics. It does, however, provide the means to manage the data in order to achieve the best effects.

Transport Control Protocol over Internet Protocol (TCP/IP)

TCP/IP was developed by the U.S. military to allow computers to communicate to each other over long distance networks. IP is responsible for moving packets of data between nodes. TCP is responsible for verifying delivery from client to server. TCP/IP forms the basis of the Internet, and is built into every common modern operating system.

User-Perceived Quality (UPQ)

A term we use to refer to the quality of a service, as provided by a network application, from the point of view of the user. It usually implies basic human perception functions, such as audio or visual perception. In order to quantify it application-specific metrics must be defined.

Video Audio Tool protocol (VAT)

VAT protocol is the data transmission protocol used by VAT, an audio conferencing application developed by the Network Research Group of Lawrence Berkeley National Laboratory.

Voice over IP (VoIP)

VoIP is a set of technologies that enables voice to be sent over a packet network. While relatively few corporations use VoIP today, private use is currently important. The global utilisation of VoIP for messaging has been growing steadily over the last years and is expected to explode in the near future.

References

[Beu-03] R. Beuran, M. Ivanovici, B. Dobinson, N. Davies, P. Thompson, “Network Quality of Service Measurement System for Application Requirements Evaluation”, *Int. Symp. on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'03)*, July 20-24, 2003, Montreal, Canada, pp. 380-387.

[Bou-02] C. Boutremans, G. Iannaccone, C. Diot, “Impact of link failures on VoIP performance”, *Proc. of the 12th Int. Workshop NOSSDAV*, Miami, Florida, USA, May 2002, pp. 63-71.

[Cav-02] V. Cavalli, E. Verharen, “TF-STREAM Real Time Multimedia Applications", *TERENA Technical Report*, March 2002.

[Con-02] A. E. Conway, Y. Zhu, “Analyzing Voice-over-IP Subjective Quality as a Function of Network QoS: A Simulation-Based Methodology and Tool”, *Lecture Notes in Computer Science 2324*, Springer-Verlag, 2002, pp. 289-308.

[Dem-02] C. Demichelis, P. Chimento, “IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)”, *IETF RFC 3393*, November 2002.

[God-00] D. Goderis (editor), “Functional Architecture Definition and Top Level Design”, *TEQUILA Project*, September 2000.

[Gri-00] D. Griffin (editor), “Selection of Simulators, Network Elements and Development Environment and Specification of Enhancements”, *TEQUILA Project*, May 2000.

[Gün-01] E. Gündüzhan, K. Momtahan, “A Linear Prediction Based Packet Loss Concealment Algorithm for PCM Coded Speech”, *IEEE Trans. On Speech and Audio Processing*, Vol. 9, No. 8, November 2001, pp. 778-785.

[G.107] ITU-T Recommendation G.107, “The E-model, a computational model for use in transmission planning”, *ITU-T*, May 2000.

[G.711] ITU-T Recommendation G.711, “Pulse Code Modulation (PCM) of voice frequencies”, *ITU-T*, 1993.

[G.726] ITU-T Recommendation G.726, “40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)”, *ITU-T*, 1990.

[G.729] ITU-T Recommendation G.729, “Coding of speech at 8 kbit/s using Conjugate-Structure Algebraic-Code-Excited Linear-Prediction (CS-ACELP)”, *ITU-T*, March 1996.

[Int2] Internet2 End-to-End Performance Initiative, http://e2epi.internet2.edu/.

[I.380] ITU-T Recommendation I.380, “Internet Protocol (IP) Data Communication Service – IP Packet Transfer and Availability Performance Parameters”, *ITU-T*, February 1999.

[Jia-03] W. Jiang, H, Schulzrinne, “Assessment of VoIP Service Availability in the Current Internet”, *Passive and Active Measurement Workshop (PAM)*, La Jolla,California, USA, April 6-8, 2003, pp. 151-157.

[Koo-02] R. Koodli, R. Ravikanth, “One-way Loss Pattern Sample Metrics”, *IETF RFC 3357*, August 2002.

[Malden] Malden Electronics Ltd., *http://www.malden.co.uk.*

[Man-01] D. Manikis (editor), “Overview of the TEQUILA Reference Testbeds”, *TEQUILA Project*, February 2001.

[Mar-02] A. Markopoulou, F. Tobagi, M. Karam, “Assessment of VoIP quality over Internet backbones”, *Proc. of IEEE Infocom*, New York, USA, June 23-27, 2002, pp. 150-159.

[Mir-02] Miras, D. 2002. "A Survey on Network QoS Needs of Advanced Internet Applications", Internet2 QoS Working Group, December.

[Moo-98] S. B. Moon, J. Kurose, D. Towsley, “Packet Audio Playout Delay Adjustment: Performance Bounds and Algorithms”, *ACM/Springer Multimedia Systems*, Vol. 6, January 1998, pp. 17-28.

[NIST] National Institute of Standards and Technology, NIST Net network emulator, *http://snad.ncsl.nist.gov/itg/nistnet.*

[PAMS] Malden Electronics Ltd., “PAMS – A Perceptual Analysis/Measurement System”, *http://www.malden.co.uk/products/dsla/pams.htm.*

[P.800] ITU-T Recommendation P.800, “Methods for subjective determination of transmission quality”, *ITU-T*, August 1996.

[P.861] ITU-T Recommendation P.861, “Objective quality measurement of telephone-band (300-3400 Hz) speech codecs”, *ITU-T*, February 1998.

[P.862] ITU-T Recommendation P.862, “Perceptual evaluation of speech quality (PESQ), an objective method for end to end speech quality assessment of narrow-band telephone networks and codecs”, *ITU-T*, February 2001.

[Rei-**] V. Reijs, “Perceived Quantitative Quality of Applications”, *http://www.heanet.ie/Heanet/projects/nat_infrastruct/perceived.html.*

[Rah-93] M. Rahnema, “Overview of the GSM system and protocol architecture”, *IEEE Communications Magazine*, April 1993.

[Sch-03] H. Schulzrinne, S. Casner, “RTP Profile for Audio and Video Conferences with Minimal Control”, *IETF RFC 3551*, July 2003.

[Ser-01] V. Servis, “Measuring speech quality over VoIP networks”, *The TOLLY Group*, December 2001.

[Tei-99] B. Teitelbaum (editor), “QBone Architecture”, *Internet2 QoS Working Group*, draft, August 1999.

[Wah-00] B. W. Wah, X. Su, D. Lin, “A survey of error-concealment schemes for real-time audio and video transmissions over the Internet”, *Proc. IEEE Int. Symp. Multimedia Software Engineering*, Taipei, Taiwan, December 2000, pp. 17-24.

[Wil-**] B. C. Wiles, J. Walker, Speak Freely, *http://www.speakfreely.org.*

[Y.1541] ITU-T Recommendation Y.1541, “Network performance objectives for IP-based services”, *ITU-T*, May 2002.